СОДЕРЖАНИЕ

[Введение 4](#_Toc66486956)

[1. Изучение предметной области 6](#_Toc66486957)

[1.1. Анализ аналогов 6](#_Toc66486958)

[1.2. Выявление требований к функционалу программного средства 8](#_Toc66486959)

[1.3. Анализ проблемы большого объема данных видеофайлов 9](#_Toc66486960)

[1.4. Возможные решения проблемы большого объема данных видеофайлов. 10](#_Toc66486961)

[1.5. Постановка задачи 11](#_Toc66486962)

[1.6. Источники входных данных 12](#_Toc66486963)

[1.7. Выходная информация 12](#_Toc66486964)

[1.8. Требования к программе 12](#_Toc66486965)

[1.9. Требования к программной документации 13](#_Toc66486966)

[1.10. Стадии и этапы разработки 13](#_Toc66486967)

[2. Постановка задачи и Проектирование программного средства 14](#_Toc66486968)

[2.1. Назначение и область применения программного средства 14](#_Toc66486969)

[2.2. Выбор предполагаемых средств разработки 14](#_Toc66486970)

[2.3. Проектирование структуры 15](#_Toc66486971)

[2.4. Проектирование средствами UML 16](#_Toc66486972)

[3. Реализация. ОпиСание алгоритмов работы программного средства 21](#_Toc66486973)

[3.1. Обоснование выбранных основных средств разработки 21](#_Toc66486974)

[3.2. Описание готовых использованных элементов 23](#_Toc66486975)

[3.3. Описание технологии протокола авторизации OAuth 2.0 25](#_Toc66486976)

[3.4. Описание проблем при реализации облачного сохранения 28](#_Toc66486977)

[4. Тестирование 29](#_Toc66486978)

[4.1. Функциональное тестирование 29](#_Toc66486979)

[4.2. Тестирование с помощью тест-кейсов 31](#_Toc66486980)

[5. Экономиечский раздел 35](#_Toc66486981)

[5.1. Исходные данные 35](#_Toc66486982)

[5.2. Определение себестоимости программного обеспечения (ПО), как базы для формирования цены 35](#_Toc66486983)

[5.3. Расчет затрат на материалы 36](#_Toc66486984)

[5.4. Обоснование трудоемкости разработки ПО 37](#_Toc66486985)

[5.5. Расчет затрат на оплату труда 39](#_Toc66486986)

[5.6. Расчет полной себестоимости программного обеспечения 40](#_Toc66486987)

[Список использованных источников 43](#_Toc66486988)

[Заключение 44](#_Toc66486989)

[Приложение 1. ИСХОДНЫЙ ТЕКСТ ПРОГРАММНОГО СРЕДСТВА 45](#_Toc66486990)

[Приложение 2. БЛОК-СХЕМА АЛГОРИТМА РАБОТЫ ПРОГРАММНОГО СРЕДСТВА 55](#_Toc66486991)

[Приложение 3. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 56](#_Toc66486992)

[Приложение 4. ПРОТОКОЛ ИСПЫТАНИЙ ПРОГРАММНОГО СРЕДСТВА 61](#_Toc66486993)

Введение

Все больше сфер человеческой деятельности перестраиваются под работу за компьютером и использование видеоинформации, связанной с демонстрацией последовательности действий для управления операционной системой компьютера или программ, влечет за собой потребность в аппаратно-программном и программном обеспечении для выполнения экранной видеозаписи.

Для решения различного рода задач, таких как обучение использованию программного обеспечения с наглядной демонстрацией действий пользователя при работе с операционной системой, запись игрового процесса компьютерных игр, создание видео-презентаций и других целей, используются средства для сохранения информации о действиях пользователя за компьютером в виде цифровой видеозаписи.

Стоит учесть, что данные в видео формате имеют достаточно большой объем что приводит к отказу от данной формы представления информации при частом использовании данного подхода. Как пример решения данной проблемы, применяется хранение данных на различных серверах облачных сервисов или развитие технологий сжатия видеозаписи без видимых потерь качества.

Сохранить процесс выполнения действий, происходящих на экране компьютера способно программное обеспечение для захвата видео с экрана монитора. Подобное программное обеспечение применятся в различных сферах таких как образование, бизнес, индустрия развлечений и научные исследования в области информационных технологий. Также существуют аппаратно-программные средства такие как карата захвата, упрощающая нагрузку на рабочее устройство при записи видеоинформации.

Целью технологии захвата видео с экрана компьютера является создание цифровой видеозаписи информации, выводимой на экран компьютера.

Существует достаточно большое число программных средств для захвата видео с экрана мониторов, которые решают ряд задач таких как:

1. решение проблемы слишком больших размеров видео файлов;
2. решение проблем с оптимизацией использования ресурсов компьютера;
3. решение проблем с настройкой приложения под пользовательские нужды.

При разработке приложений для захвата видео с экрана компьютера стоит учесть вышеописанные цели и задачи.

Предполагается, что приложение, разрабатываемое в рамках данного проекта, сможет корректно выполнять основные функции, быть в достаточной степени оптимизировано для стабильной работы и иметь ряд уникальных функций таких как сохранение файлов в облачное хранилище.

Целью дипломного проекта является рассмотрение актуальных технологий для захвата видео с экрана компьютера и разработка программного средства на основании полученных знаний о данных технологиях, выявление актуальных проблем, возникающих в ходе видеозаписи.

Задачами дипломного проекта являются:

* изучить актуальные технологии для захвата видео с экрана компьютера;
* проанализировать процесс формирования видеозаписи;
* разработать программное средство;
* исследовать и реализовать способы решения проблем с оптимизацией записи видео;
* исследовать и реализовать решения проблемы с большим объёмом данных видеозаписей.

# Изучение предметной области

## Анализ аналогов

Для выявления требований к основному функционалу программного средства, разрабатываемого в рамках дипломного проекта будут рассмотрены три наиболее востребованных варианта программ для записи видео с экрана монитора.

«Экранная камера» – программа обеспечивающая наличие всех необходимых средств для произведения качественного захвата экрана. На фоне аналогов, программа демонстрирует целый спектр преимуществ. Она отлично справляется с записью онлайн-трансляций, видеороликов из различных игр, программ, фильмов и так далее. Имеет относительно удобный русскоязычный интерфейс (рис. 1).

Особенности:

* самое удобное управление среди конкурентов;
* реализация современного русскоязычного интерфейса;
* эффективное формирование обучающих роликов и презентаций;
* сохранение готового материала в любом распространенном формате;
* возможность создания записей в HD-формате;
* наличие встроенного редактора и функции, позволяющей записывать звук поверх видео;
* специальный функционал добавления эффектов, оформлений, титров, аудиозаписей и заставок.

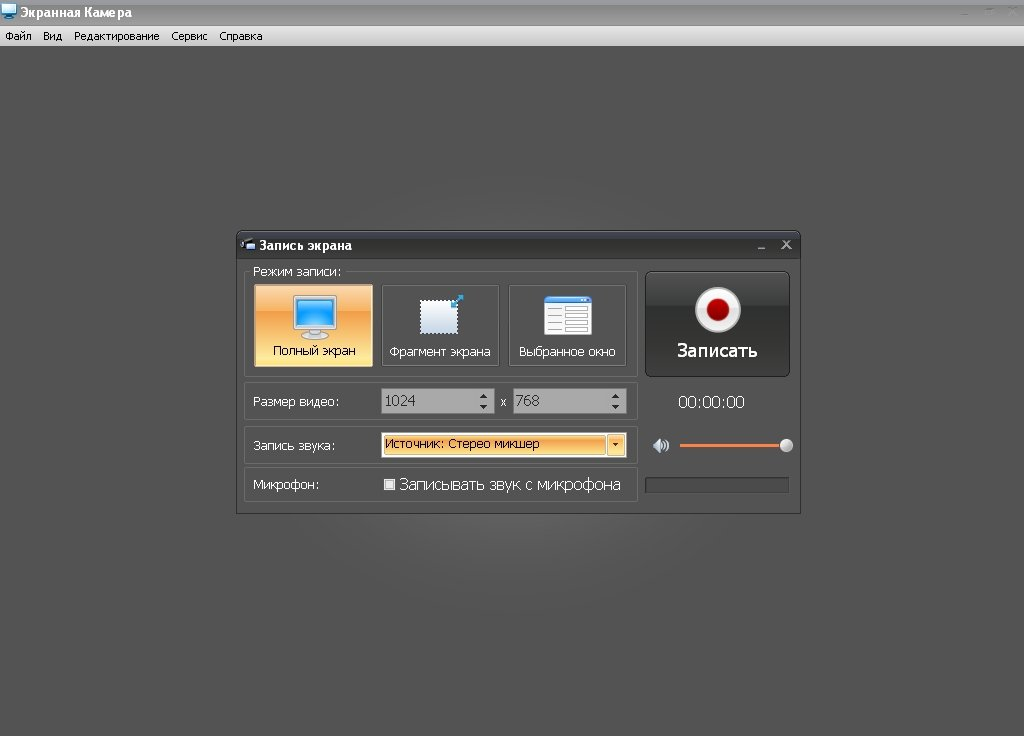


Рисунок 1.1 – Интерфейс пользователя программы «Экранная камера»

Movavi Screen Capture – данная утилита записи подойдет нескольким категориям пользователей. Преподаватели могут создавать уроки, а игроки компьютерных игр могут использовать программу для записи игрового процесса с частотой до 60 кадров/с. Имеется приложение для монтирования видеозаписей. Если просматривать видео онлайн в определенном формате (1080р, 720р и т.д.), то средство захвата сделает ролик в таком же разрешении. Имеет компактный интерфейс пользователя что позволяет облегчить взаимодействие с программным средством (рис. 2).

Особенности:

* более 180 форматов для сохранения видео;
* предустановки для записи экранного видео в формате мобильных устройств;
* добавление водяных знаков на отснятый материал;
* оптимизация работы видеокарты;
* поддерживается размер файла до 4k;
* наличие встроенного инструментария монтажа + проигрыватель;
* экспорт онлайн (публикация на YouTube, Vimeo и в соцсетях).

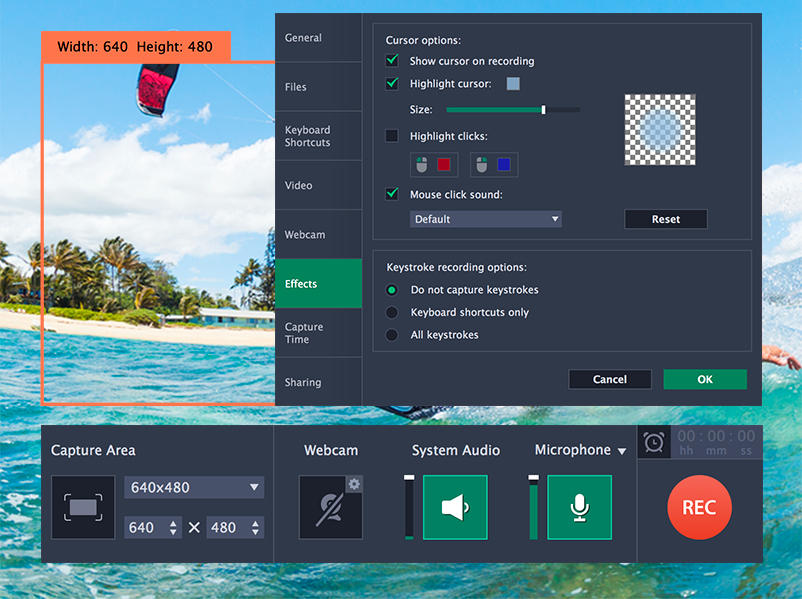


Рисунок 1.2 – Интерфейс пользователя программы «Movavi Screen Capture»

Bandicam – программа, позволяющая захватывать видео с монитора, вести запись игр, фильмов и делать скриншоты. Во время съемки, можно захватывать и определенную часть экрана, для этого достаточно выделить интересующую область. Получаемые видео и изображения компактны и при этом качественны за счет алгоритмов сжатия данных без видимых потерь качества. Изображения экрана можно получать в популярных расширениях BMP, JPEG и PNG. На фоне большинства аналогов, утилита Bandicam имеет приемлемые системные требованиями и не перегружает процессор, а также ОЗУ. Имеется возможность записать игровой процесс компьютерных игр, без помех в отличном FullHD качестве. Имеет современный интерфейс пользователя (рис. 3).

Особенности:

* произведение скриншотов и их последующее сохранение;
* работа в высоком разрешении;
* формирование компактных материалов;
* установка паузы или остановка в указанное время при необходимости;
* можно выполнять видео захват в формате AVI, приложений, основанных на технологии DirectX;
* поддержка аппаратного ускорения по технологиям AMD VCE, Intel QSV;
* работы с Xvid, MP2, MJPEG, PCM и MPEG-1.

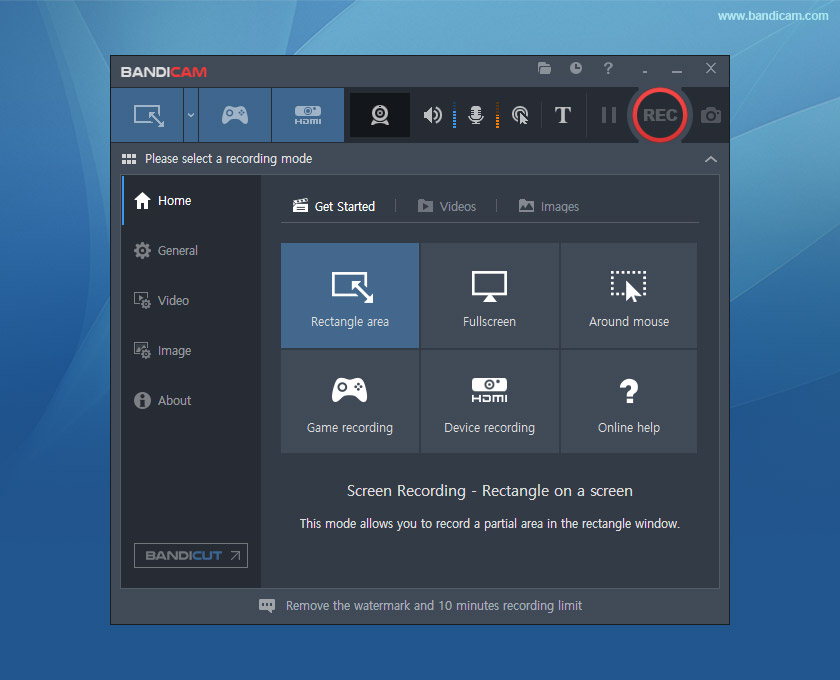


Рисунок 1.3 – Интерфейс пользователя программы «Bandicam»

## Выявление требований к функционалу программного средства

На основании анализа аналогов можно выделить следующие требования к содержанию функционала разрабатываемого программного решения:

* создание скриншотов и их последующее сохранение;
* реализация современного русскоязычного интерфейса;
* возможность создания записей в HD-формате;
* захват видео с поддержкой облегчения нагрузки на аппаратные средства.

## Анализ проблемы большого объема данных видеофайлов

При создании медиа контента существует множество факторов. На этапе, когда HD не достигает отметки высоких стандартов, и люди всегда стремятся к наилучшему результату, когда дело касается качества медиа контента. Имея это в виду, при формировании плана разработки вам необходимо учитывать размер видеофайлов, которые будут созданы, и то, как их передавать их в глобальном масштабе команде и/или аудитории.

### Продолжительность видео

При создании раскадровки для контента в первую очередь следует учитывать длину медиа файла. Этот фактор играет большую роль в определении размера файла, поскольку он является основой контента. Естественно, чем короче продолжительность, тем меньше может быть оценен размер видео.

### Разрешение Размеры видеофайлов

На данный момент определены и находятся в повсеместном использовании следующие форматы видеофайлов:

* 8K Video (7680×4320);
* 4K (Ultra HD) 3840×2160;
* 2K (2048 x 1080);
* 1080p 1920×1080;
* 720p 1280×720.

### Разрешение видео

Если необходимо большое разрешение, 8K или 4K. Существуют инструменты, которые предоставляют несколько вариантов разрешения. Решения FileCatalyst позволяют работать, не учитывая размер конечного файла, поскольку можно передавать большие видеофайлы по всему миру с более высокой скоростью. Качество видео в конечном итоге влияет на размер видеофайла разрешение экрана или выбранной области: чем выше качество растрового изображения кадра, тем больше пикселей составляет само изображение. Более высокое качество приводит к большему размеру файла, а больший размер файла приводит к большим трудностям при их передаче.

### Формат видео

Формат видео также играет большую роль в размере видеофайла. Есть два основных компонента формата видео: кодек и контейнеры. Кодеки относятся к процессу сжатия файла, а контейнер указывает, как видеофайл кодируется, и отображает это через тип расширения файла (например, .mov, .mp4, .avi и др.). Есть проблемы, которые могут возникнуть при сжатии большого видеофайла. Предположим, создано двухчасовое видео с разрешением 8K, чтобы уменьшить его размер, его необходимо сжать. В этом процессе велика вероятность потери данных. Приложения для сжатия видео работают, ища дубликаты в кадре и сохраняя их от кадра к кадру. Например, один кусочек голубого неба совпадает с другим кусочком голубого неба. Синяя секция проходит через каждый кадр. При высокой степени сжатия это становится очевидным.

Решения, устраняющие необходимость сжатия при работе над проектом, очень полезны. Поскольку полное качество остается неизменным, а время ожидания передачи практически отсутствует.

### Частота кадров видео

Наиболее быстрый и эффективный способ проанализировать размер видео – это узнать количество кадров в секунду (FPS). Чем выше FPS, тем лучше качество видео.

## Возможные решения проблемы большого объема данных видеофайлов.

Из анализа проблем большого объема данных видеофайлов можно выявить пару решений экономии памяти физического носителя при видеозаписи экрана ПК:

* сжатие видеофайлов алгоритмами сжатия без видимых потерь данных;
* сжатие видеофайлов путем изменения из разрешения на меньшее;
* ограничение на число кадров в секунду;
* ограничение на длину видео.

Все вышеописанные методы являются уменьшением размера самого видеофайла для экономии места на физическом носителе. Имеется ещё одно решение для экономии информационного пространства физического носителя, при этом не затрагивающее сам видеофайл – сохранение в облачное хранилище данных.

Облачное хранилище данных (англ. cloud storage) — модель онлайн-хранилища, в котором данные хранятся на многочисленных распределённых в сети серверах, предоставляемых в пользование клиентам, в основном, третьей стороной. В отличие от модели хранения данных на собственных выделенных серверах, приобретаемых или арендуемых специально для подобных целей, количество или какая-либо внутренняя структура серверов клиенту, в общем случае, не видна. Данные хранятся и обрабатываются в так называемом «облаке», которое представляет собой, с точки зрения клиента, один большой виртуальный сервер. Физически же такие серверы могут располагаться удалённо друг от друга географически.

Для примера будет рассмотрена программа «Редактор скриншотов в Яндекс.Диске» из приложения Яндекс диск на ПК (рис. 4).

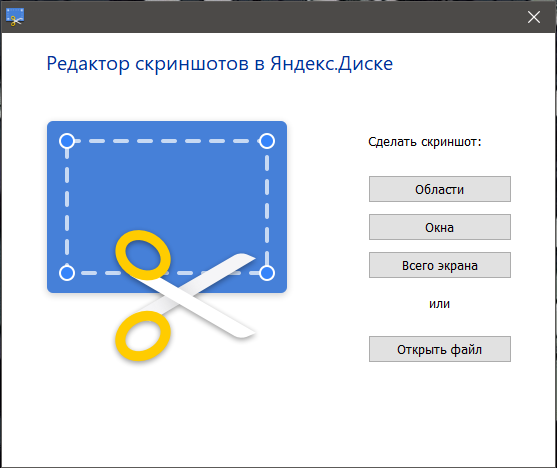


Рисунок 1.4 – Интерфейс редактора скриншотов Яндекс диска

Приложение позволяет напрямую сохранять растровые изображения в облачное хранилище, при этом не затрагивая физическое хранилище.

У облачных сервисов имеется API (Application Programming Interface) для внедрения его в разрабатываемое программное средство, что позволяет составить решение при экономии информационного пространства физического носителя. Это решение также можно использовать при записи видеофайлов.

## Постановка задачи

### Назначение и область применения

Наименование проекта – «Программное средство для захвата изображений и видео с экранов ПК». Данный проект предназначен для перевода информации, выводимой на экран компьютера в цифровой формат в виде видеофайла или растрового изображения.

Имеет место применение подобного рода программных средств для таких целей или процессов как:

1. наглядное обучение сотрудников или студентов пользованию программному обеспечением той или иной системы;
2. запись игрового процесса компьютерных игр или процесса рисования в графических редакторах в качестве развлекательного контента;
3. создание презентаций.

### Основание для разработки

Тема «Программное средство для захвата изображений и видео с экранов ПК» выдана и утверждена руководителем дипломного проекта.

## Источники входных данных

Исходными данным является массив байт, получаемый из информации отображения экрана ПК, системные сигналы клавиатуры и компьютерной мыши и текстовая информация вводимая в текстовые поля приложения.

## Выходная информация

Выходной информацией являются файлы растровых изображений и видеозаписей. Так же пользовательские данные в виде логина и хешированного пароля сохраняемой в базе данных приложения и сохраненные пользовательские настройки.

## Требования к программе

### Требования к функциональным характеристикам

Разрабатываемое программное средство должно корректно, без сильных затрат ресурсов ПК захватывать видео и изображения экрана монитора, экономить информационное пространство физического носителя, иметь возможность облачного сохранения файлов.

### Требования к составу и параметрам технических средств

Требования к техническим характеристикам ПК пользователя:

Минимальная конфигурация:

* тип процессора: Процессор AMD Athlon X4 845 [AD845XACI43KA]/Процессор Intel Core 2 Duo E8400;
* объём ОЗУ: 6 Гб;
* объём свободного места на жёстком диске: 60 Мб.

Рекомендуемая конфигурация:

* тип процессора: Процессор Intel Core i3-9100T/Процессор AMD Ryzen 3 Pro 3200G;
* объём ОЗУ: 8 Гб;
* объём свободного места на жёстком диске: 60 Мб.

### Требования к информационной и программной совместимости

Программа должна работать под управлением семейства операционных систем Win 32/64, серия Windows не ниже Windows 7.

## Требования к программной документации

Состав программной документации должен включать в себя:

* техническое задание;
* экономическая часть;
* программу и методики испытаний;
* руководство пользователя.

## Стадии и этапы разработки

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

1. разработка программы;
2. разработка программной документации;
3. испытания программы.

# Постановка задачи и Проектирование программного средства

## Назначение и область применения программного средства

Программа, разрабатываемая в рамках дипломного проекта, будет предназначена для создания скриншотов и захвата видео с экрана монитора.

О возможности осуществления видео захвата экрана известно давно. Чаще используется термин скринскастинг, то есть цифровая аудио и видеозапись, которая производится непосредственно с монитора компьютера.

Как правило, этот инструмент используется для более наглядного и динамичного представления того или иного цифрового приложения.

Скринкасты делят на ряд жанров, большинство которых можно использовать в образовании, прежде всего, в дистанционном обучении:

1. демонстрация работы компьютерной программы, интернет-сервиса, обучающей программы. Задача такого видеоролика – показать привлекательность, эффективность, полезной такой программы;
2. демонстрация последовательности действий с программой, приложением. Видео инструкция о алгоритме действий;
3. обзор программного обеспечения, электронного учебного пособия, любого цифрового образовательного ресурса. В данном случае используется, как скриншот, сопровождаемый текстом, так и скринкаст, который делает обзор более наглядным и динамичным. Иногда эти два приёма сочетаются в одном ролике;
4. интерактивная доска. Учитель использует экран компьютера для трансляции одному или нескольким ученикам видео урока, в котором он играет активную роль своими комментариями, подчёркиваниями, демонстрациями медиа файлов.

Также подобного рода программы средства распространены среди людей, создающих развлекательные видеоролики, такие как записи игрового процесса компьютерной игры, записи процесса рисования в графическом редакторе и другое.

Приложения для скринкастинга используются для бизнеса, как пример, использования средств видео захвата экрана ПК для создания видео презентаций.

## Выбор предполагаемых средств разработки

В качестве целевой платформы из раздела 1, выявлено что программа должна работать под управлением семейства операционных систем Win 32/64, серия Windows не ниже Windows 7.

В качестве основных языка программирования и среды разработки подходят язык программирования C# и среда разработки Visual Studio 2019.

У языка программирования C# выделяют ряд преимуществ:

* поддержка подавляющего большинства продуктов Microsoft;
* бесплатность ряда инструментов для небольших компаний и некоторых индивидуальных разработчиков – Visual Studio, облако Azure, Windows Server, Parallels Desktop для Mac Pro и др;
* типы данных имеют фиксированный размер (32-битный int и 64-битный long), что повышает «мобильность» языка и упрощает программирование, так как вы всегда знаете точно, с чем вы имеете дело;
* автоматическая «сборка мусора». Это значит, что нам в большинстве случаев не придётся заботиться об освобождении памяти. Общеязыковая среда CLR (Common Language Runtime) сама вызовет сборщик мусора и очистит память;
* большое количество «синтаксического «сахара» – специальных конструкций, разработанных для понимания и написания кода. Они не имеют значения при компиляции;
* низкий порог вхождения. Синтаксис C# имеет много схожего с другими языками программирования, благодаря чему облегчается переход для программистов. Язык C# часто признают наиболее понятным и подходящим для новичков;
* имеется встроенная библиотека TPL (Task Parallel Library) которая позволяет параллельно выполнять сразу несколько процессов, поскольку все современные ЭВМ имеют больше одного ядра в процессоре;
* C# подходит для написания программ и приложений для таких операционных систем, как iOS, Android, MacOS и Linux.

У языка программирования C# ориентировочной платформой является семейства операционных систем Windows которая и является целевой платформой для данного проекта.

## Проектирование структуры

В качестве ключевой функции программного средства по теме «Программное средство для захвата изображений и видео с экранов ПК» была выбрана реализация облачного сохранения записанных видео и созданных скриншотов экрана ПК. Для этого будут использованы API (application programming interface) облачных сервисов такие как: Яндекс Диск, Google Drive, DropBox.

При отсутствии стабильного подключения и корректной передачи большего объема данных, будет использован API для Microsoft Azure Blob Storage, поскольку данное облачное хранилище предназначено для хранения больших объемов данных и само хранилище распологается на управляемом разработчиком сервере что упрощает сопровождение программного средства.

Вышеописанные сервисы имеют API подходящий для целевого языка программирования.

API – описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой.

Приложение будет подразделено на два модуля:

1. пользовательский (далее UI) – модуль, отвечающий за сохранение настроек пользователя и отображение графического интерфейса;
2. функциональный (далее Functional) – модуль, отвечающий за работу основных функций программного средства.

Функциональный блок включает в себя блоки:

* + захват экрана (далее Screen Capture) который разделен на две части: скриншот (далее Screenshot) и скринкаст (далее Screencast) для захвата изображений и видео с экрана соответственно;
  + облачный (далее Cloud) реализация функционала API облачных хранилищ.

## Проектирование средствами UML

Для понимания возможностей пользователя во взаимодействии с программным средством, доступный пользователю функционал будет отражён на диаграмме вариантов использования (Use Case).

Сценарий использования, вариант использования, прецедент использования (англ. use case) в разработке программного обеспечения и системном проектировании это описание поведения системы, когда она взаимодействует с кем-то (или чем-то) из внешней среды.

В случае рассмотрения проектируемого программного средства выбор в основном сосредоточен на выборе между технологиями захвата, настройками пользователя и выбором места сохранения файлов.

Use Case диаграмма отображена на рисунке 2.1.

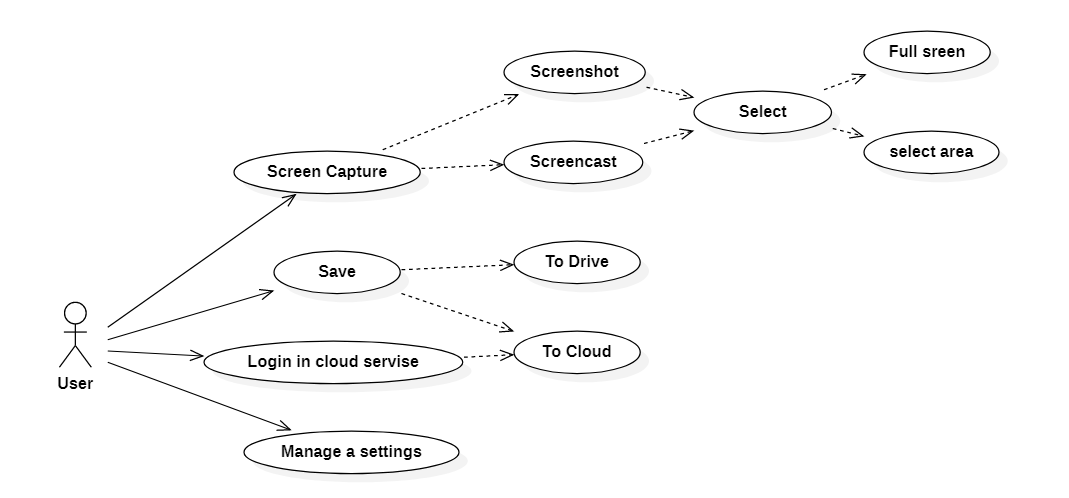


Рисунок 2.1 – Диаграмма вариантов использования

Для детального разбора структуры проектируемого программного средства составлена объектная модель с помощью нотации UML используя диаграмму классов (Class diagram).

Диаграмма классов (англ. class diagram) – структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними.

В качестве целевой системы классов приставлено средство захвата экрана с возможностью сохранения видеофайлов в облачное хранилище через подключение к облачным сервисам чрез их API.

Следовательно, необходим ряд сущностей для осуществления всех необходимых взаимодействий с программным средством такие как: классы и библиотеки, реализующие процесс захвата экрана, класс-клиент для реализации API облачных сервисов, класс контроллер для управления сохранением видеофайлов на облачный сервер, и классы для пользовательского интерфейса.

Выбранные сущности отображены в диаграмме классов со стандартами сервисами (см. рисунок 2.2).

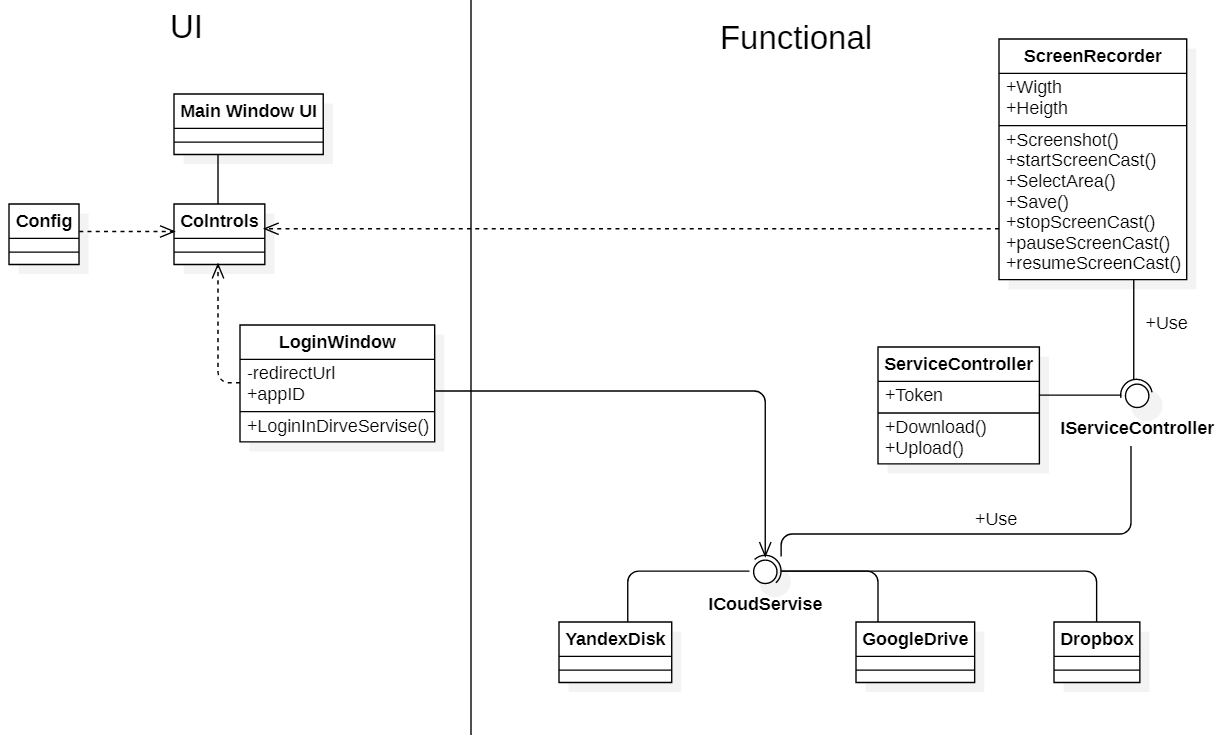


Рисунок 2.2 – Диаграмма классов со стандартными сервисами

В случае невозможности использования стандартных сервисов следует спроектировать модель где будет одно ключевое хранилище Microsoft Azure Blob Storage.

Для того чтобы определить и разграничить пользовательское пространство на облачном сервере необходима база данных, хранящая информацию о пользователях. Следовательно, необходимо учесть возможность регистрации новых пользователей.

Схема базы данных оформлена в самой диаграмме классов (рис. 2.3) поскольку будет разработана в самом приложении. Как видно на рисунке 2.3, необходима также сущность, отвечающая за управление облачным хранилищем.

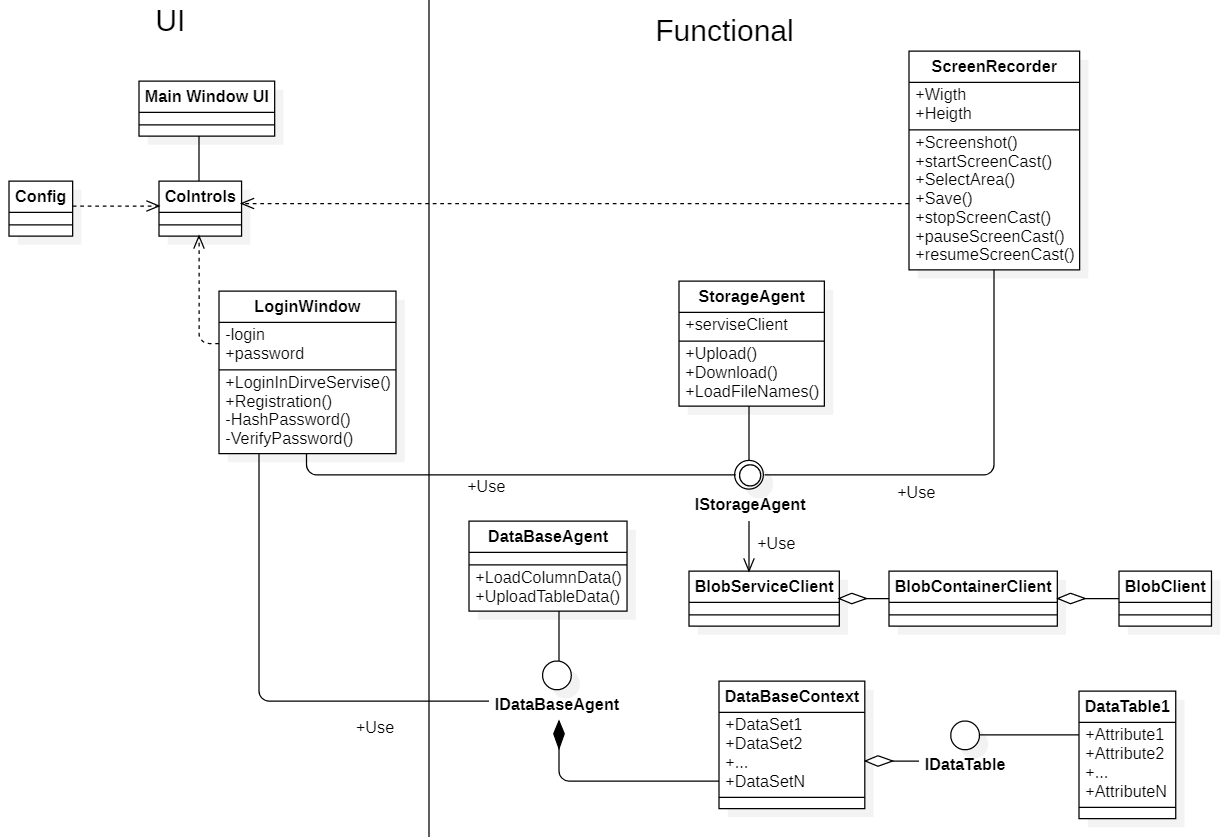


Рисунок 2.3 – Диаграмма классов с сервисом Microsoft Azure Blob Storage

Для отражения сценария действий, происходящих в процессе первого запуска проектируемого программного средства использована диаграмма последовательности (sequence diagram).

Диаграмма последовательности (англ. sequence diagram) – UML-диаграмма, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл объекта (создание-деятельность-уничтожение некой сущности) и взаимодействие актеров (действующих лиц) информационной системы в рамках прецедента.

На диаграмме последовательности (см. рисунок 2.4) отражен процесс первого запуска приложения.

Сначала пользователь активирует исполняемый файл, затем компонент пользовательского интерфейса перелает сигнал передается на контроллер и направляет команду классу User agent который определяет авторизацию пользователя в облачном сервисе и выберет папку сохранения (по умолчанию) и ответ об успешной/не успешной авторизации передается пользователю через диалоговое окно, затем пользователь может начать запись экрана и сохранить его на диск или в облако (в зависимости от успеха авторизации).

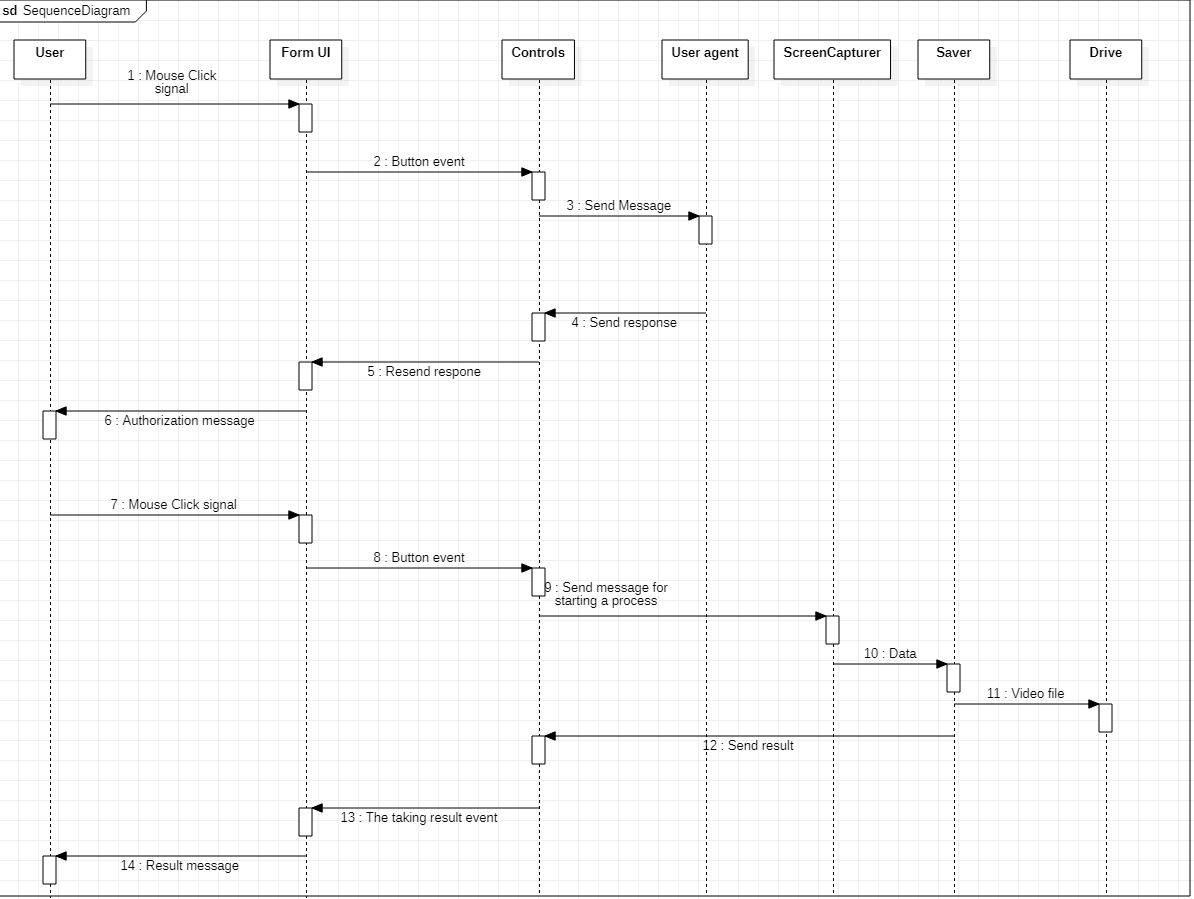


Рисунок 2.4 – Диаграмма последовательности

Для рассмотрения движения потока управления и того как он переходит от одной деятельности к другой построена диаграмма активности (Activity diagram) отражающая динамические аспекты поведения системы.

Диаграмма деятельности (англ. activity diagram) – UML-диаграмма, на которой показаны действия, состояния которых описано на диаграмме состояний. Под деятельностью (англ. activity) понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчинённых элементов — вложенных видов деятельности и отдельных действий англ. action, соединённых между собой потоками, которые идут от выходов одного узла ко входам другого.

Диаграмма отображена на рисунке 2.5.

На диаграмме деятельности (рисунок 2.5) отражены процессы которые могут происходить при взаимодействии с программным средством. Начинается с запуска приложения далее идет определение действий пользователя. Если пользователь авторизовался в облачном сервисе, то сохранение идет в облако, если авторизация не удалась или пользователь от нее отказался тогда сохранение будет на физический носитель пользователя.

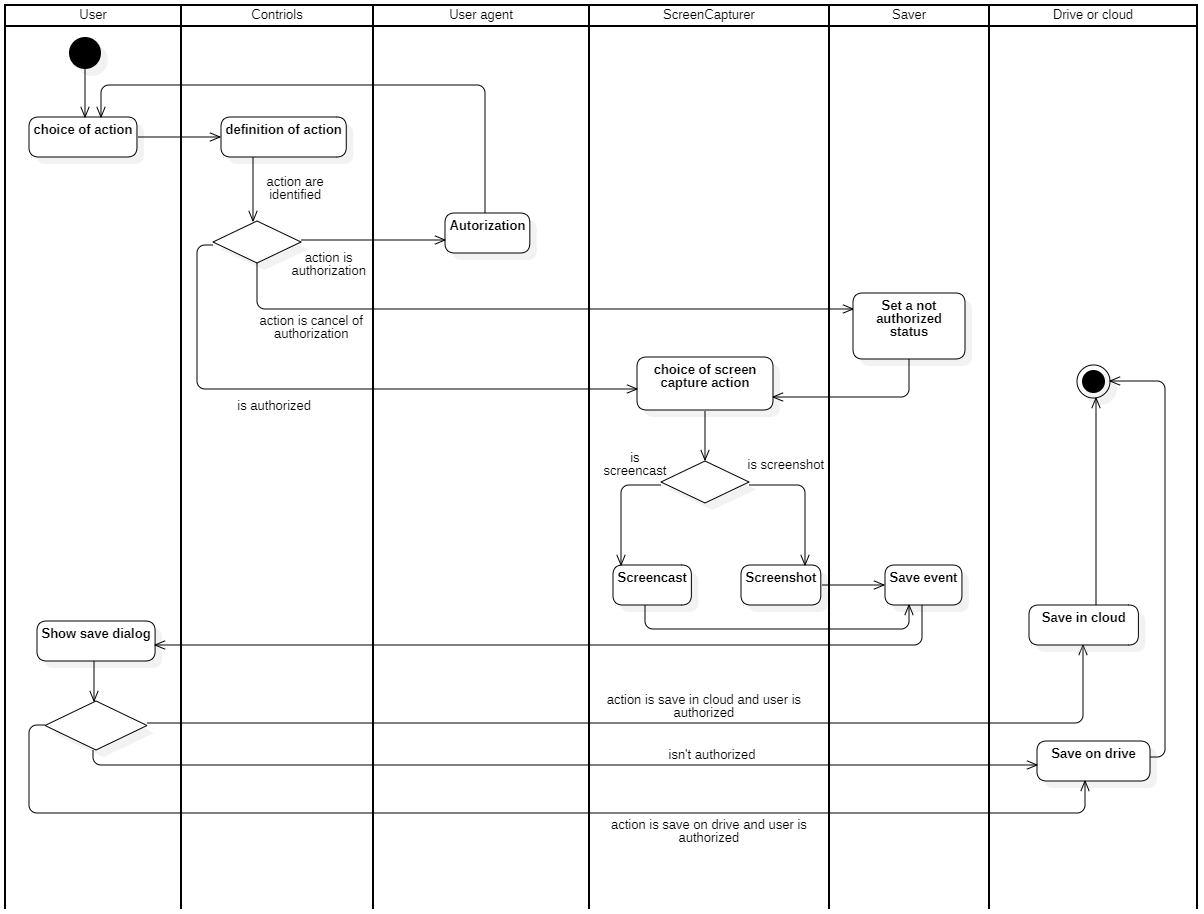


Рисунок 2.5 – Диаграмма деятельности

# Реализация. Описание алгоритмов работы программного средства

## Обоснование выбранных основных средств разработки

### Среда разработки Visual Studio

Visual Studio является одним из самых мощных и распространённых инструментов разработки. Среда разработки представляет много инструментов для работы с кодом.

В комплект входят следующие основные компоненты:

1. Visual Basic.NET - для разработки приложений на VisualBasic;
2. Visual C++ - на традиционном языке C++;
3. Visual C# - на языке C# (Microsoft);
4. Visual F# - на F# (Microsoft Developer Division).

Функциональная структура среды включает в себя:

* редактор исходного кода, который включает множество дополнительных функций, как авто дополнение IntelliSense, рефракторинг кода и т. д.;
* отладчик кода;
* редактор форм, предназначенный для упрощённого конструирования графических интерфейсов;
* веб-редактор;
* дизайнер классов;
* дизайнер схем баз данных.

Visual Studio также позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (Subversion и VisualSourceSafe), добавление новых наборов инструментов (для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования или инструментов для прочих аспектов процесса разработки программного обеспечения).

Интегрированная среда разработки (IntegratedDevelopmentEnvironment - IDE) Visual Studio предлагает ряд высокоуровневых функциональных возможностей, которые выходят за рамки базового управления кодом.

Ниже перечислены основные преимущества IDE-среды Visual Studio.

Меньше кода для написания. Для создания большинства приложений требуется большое количество стандартного стереотипного кода. Например, добавление Web-элемента управления, присоединение обработчиков событий и корректировка форматирования требует установки в разметке страницы ряда деталей. В Visual Studio такие детали устанавливаются автоматически.

Более высокая скорость разработки. Многие из функциональных возможностей Visual Studio направлены на то, чтобы помогать разработчику делать свою работу как можно быстрее. Удобные функции, вроде функции IntelliSense (которая умеет перехватывать ошибки и предлагать правильные варианты), функции поиска и замены (которая позволяет отыскивать ключевые слова как в одном файле, так и во всем проекте) и функции автоматического добавления и удаления комментариев (которая может временно скрывать блоки кода), позволяют разработчику работать быстро и эффективно.

Возможности отладки. Предлагаемые в Visual Studio инструменты отладки являются наилучшим средством для отслеживания загадочных ошибок и диагностирования странного поведения. Разработчик может выполнять свой код по строке за раз, устанавливать интеллектуальные точки прерывания, при желании сохраняя их для использования в будущем, и в любое время просматривать текущую информацию из памяти.

Visual Studio также имеет и множество других функций: возможность управления проектом; встроенная функция управления исходным кодом; возможность рефакторизации кода; мощная модель расширяемости. Более того, в случае использования Visual Studio Team System разработчик получает расширенные возможности для модульного тестирования, совместной работы и управления версиями кода.

### Целевая платформа .NET Framework

Платформа Microsoft .NET Framework – это компонент Microsoft Windows, который пользователи могут получить с помощью различных каналов распространения, в число которых входят веб-узел Центра обновления Майкрософт и центр загрузки Microsoft. Кроме того, этот компонент устанавливается на некоторые новые компьютеры. В этой статье описаны преимущества, обеспечиваемые установкой платформы .NET Framework

В приведенном ниже списке перечислены некоторые преимущества, которые дает установка платформы .NET Framework.

Более простая установка коммерческих приложений. Для поддержки основных функций многих коммерческих приложений корпорации Майкрософт и сторонних производителей необходимо наличие на компьютере платформы .NET Framework. Если она установлена, установить эти приложения будет проще.

Более простая установка обновлений. После установки платформы .NET Framework она автоматически обновляется с помощью службы Центра обновления Майкрософт. Если автоматическое обновление выполняется с помощью панели управления, то для всех установленных на компьютере версий .NET Framework всегда будут устанавливаться последние пакеты обновления.

Ниже перечислены некоторые новые возможности платформы .NET Framework 3.5:

* интеграция WCF и WF;
* службы WCF, предназначенные для долговременного использования;
* поддержка частично доверительного отношения для WCF в определенных сценариях;
* модель веб-программирования WCF (JSON);
* синдикация WCF (Atom и RSS);
* улучшенные правила WF;
* взаимодействие WCF с обновленными стандартными приложениями обозревателя XAML (XBAP);
* интерактивный трехмерный API и трехмерный API с поддержкой двухмерного режима в WPF;
* улучшенная интеграция приложений WPF XBAP с обозревателями, которая позволяет объединить доступ к файлам cookie;
* модель надстроек пользовательского интерфейса WPF;
* Поддержка Firefox для приложений XBAP.

### Система для построения клиентских приложений WPF

Платформа Windows Presentation Foundation (WPF) позволяет создавать клиентские приложения для настольных систем Windows с привлекательным пользовательским интерфейсом.

В основе WPF лежит независимый от разрешения векторный модуль визуализации, использующий возможности современного графического оборудования. Возможности этого модуля расширяются с помощью комплексного набора функций разработки приложений, которые включают в себя язык XAML, элементы управления, привязку к данным, макет, двумерную и трехмерную графику, анимацию, стили, шаблоны, документы, мультимедиа, текст и типографические функции. WPF является частью .NET, поэтому имеется возможность создавать приложения, включающие другие элементы .NET API.

XAML — это язык разметки на основе XML, который служит для определения внешнего вида приложения в декларативной форме. Обычно он используется для создания окон, страниц и пользовательских элементов управления, а также их заполнения элементами управления, фигурами и графическими элементами.

Основанием для выбора данной платформы является целевая платформа данного дипломного проекта.

## Описание готовых использованных элементов

### Библиотека Accord.Video.FFMPEG

Accord.Video.FFMPEG – Содержит классы и методы для обработки видеоисточников с помощью FFMPEG. Этот пакет создан на основе AForge.NET Framework и является частью Accord.NET Framework.

Данный пакет имеет функции покадрового создания видео из множества изображений. К примеру, скриншотов экрана для создания видеозаписи процесса работы за компьютером.

### Библиотека CSScreenRecorder

CSScreenRecorder - это библиотека .NET, которая позволяет фиксировать активность экрана ПК в видеофайл AVI (стандартный или сжатый) с помощью C # и VB .NET.

Библиотека содержит встроенный инструмент определения области, который позволяет выбрать область экрана для записи, а также можно получить положение и размер активного окна.

Кроме того, имеется возможность легко установить видеокодек и аудиокодек видеофайла, установить качество, установить отображение курсора мыши и многое другое, всего лишь с несколькими строками исходного кода.

Библиотека имеет ряд преимуществ:

1. библиотека .NET (стандартная) для .NET Framework и .NET Core;
2. захват экрана в файлы AVI (Audio Video Interleave);
3. стандартный AVI без сжатия или с компрессией. например:
   * сжатие JPEG;
   * кодек Xvid MPEG-4.
4. настраиваемое качество видео.
5. запись звук с микрофона / динамиков;
6. WaveIn / WASAPI / WASAPI Loopback;
7. стандартное сжатие PCM WAV или MP3;
8. установить битрейт / частоту кадров / каналы / битовую глубину звука;
9. отключить звук;
10. отслеживание действия мыши (необязательно);
11. получить предварительный просмотр записанных кадров;
12. запись остановка, пауза и возобновление записи;
13. установить ширину и высоту для захвата;
14. установить левое и верхнее положение для захвата;
15. установить частоту кадров (FPS);
16. встроенный инструмент выделения:
    * выбрать область для записи;
    * выбрать цвет и внешний вид рамки выделения;
    * выбрать положение и размер активного окна.
17. качественный результат;
18. получить предварительный просмотр результатов;
19. добавить водяной знак в видео файл;
20. установка положения водяного знака;
21. установка размер шрифта водяного знака;
22. установка цвет водяного знака;
23. походит для приложений WPF / Winform / Console на базе .NET;
24. много актуальных событий;
25. встроенная обработка потоков;
26. безопасный выход;
27. может работать как библиотека;
28. легко использовать;
29. для C # / VB / ASP .NET;
30. C # / VB .NET хорошо документированные примеры.

### API Google диска

API Google Диска позволяет создавать приложения, использующие облачное хранилище Google Диска. Данный API позволяет разрабатывать приложения, которые интегрируются с Google Диском, и создавать надежные функциональные возможности в своем приложении с помощью Google Drive API.

API Google диска можно использовать, чтобы:

* скачивание файлов с Google Диска и загрузка файлы на Google Диск;
* поиск файлов и папков, хранящиеся на Google Диске. Создание сложных поисковых запросов, которые возвращают любое из полей метаданных файла в ресурсе Files;
* разрешение пользователям обмениваться файлами, папками и дисками для совместной работы над контентом;
* Возможность совместить с Google Picker API для поиска всех файлов на Google Диске, затем возвращая имя файла, URL-адрес, дату последнего изменения и пользователя;
* создание сторонних ярлыков, которые представляют собой внешние ссылки на данные, хранящиеся за пределами Диска, в другом хранилище данных или облачной системе хранения;
* создание специальной папки на Диске для хранения данных вашего приложения, чтобы приложение не могло получить доступ ко всему пользовательскому контенту, хранящемуся на Google Диске. См. Раздел Хранение данных для конкретного приложения;
* Интегрирование с пользовательским интерфейсом Google Диска, который представляет собой стандартный веб-интерфейс Google, который вы можете использовать для взаимодействия с файлами Диска. Чтобы узнать все, что вы можете делать с приложением Диска, которое вы интегрируете с пользовательским интерфейсом Google Диска.

## Описание технологии протокола авторизации OAuth 2.0

Данный протокол используется в API облачных хранилищах, которые применяются в программном средстве дипломного проекта, по этой причинена уместно рассмотреть данный протокол в подробностях.

Возможности версии 1.0 данного протокола рассмотрены не будут.

OAuth 2.0 – протокол авторизации, позволяющий выдать одному сервису (приложению) права на доступ к ресурсам пользователя на другом сервисе. Протокол избавляет от необходимости доверять приложению логин и пароль, а также позволяет выдавать ограниченный набор прав, а не все сразу.

### Принцип работы OAuth 2.0

Как и первая версия, OAuth 2.0 основан на использовании базовых веб-технологий: HTTP-запросах, перенаправления и т. п. Поэтому использование OAuth возможно на любой платформе с доступом к интернету и браузеру: на сайтах, в мобильных и desktop-приложениях, плагинах для браузеров и др.

Ключевое отличие от OAuth 1.0 – простота. В новой версии нет громоздких схем подписи, сокращено количество запросов, необходимых для авторизации.

Общая схема работы приложения, использующего OAuth, такова:

1. получение авторизации;
2. обращение к защищенным ресурсам.

Результатом авторизации является access token – криптографический ключ, предъявление которого является пропуском к защищенным ресурсам. Обращение к ним в самом простом случае происходит по HTTPS с указанием в заголовках или в качестве одного из параметров полученного access token

В протоколе описано несколько вариантов авторизации, подходящих для различных ситуаций:

* авторизация для приложений, имеющих серверную часть (чаще всего, это сайты и веб-приложения);
* авторизация для полностью клиентских приложений (мобильные и desktop-приложения);
* авторизация по логину и паролю;
* восстановление предыдущей авторизации.

### Авторизация для приложений, имеющих серверную часть

Авторизация для приложений, имеющих серверную часть, отражена на рисунке 3.1.

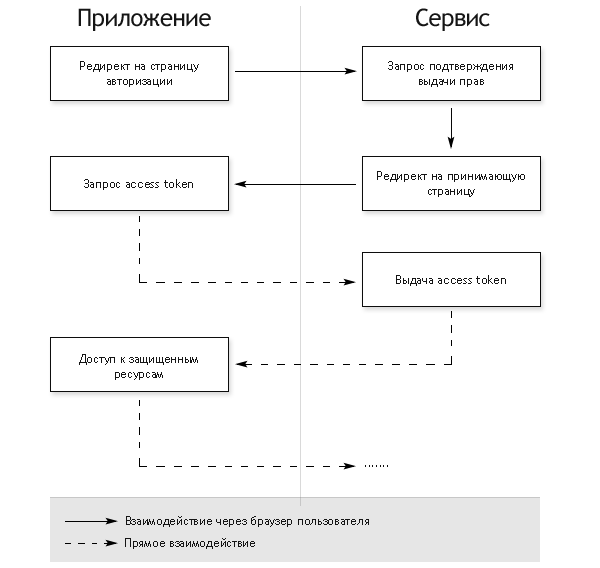


Рисунок 3.1 - Авторизация для приложений, имеющих серверную часть

Редирект (перенаправление, переадресация, форвардинг) – это способ, который позволяет один и тот же документ/web ресурс сделать доступным с других адресов (URL).

Примерный алгоритм работы:

1. редирект на страницу авторизации;
2. на странице авторизации у пользователя запрашивается подтверждение выдачи прав;
3. в случае согласия пользователя, браузер перенаправляется на URL, указанный при открытии страницы авторизации, с добавлением в GET-параметры специального ключа – authorization code;
4. Сервер приложения выполняет POST-запрос с полученным authorization code в качестве параметра. В результате этого запроса возвращается access token.

Это самый сложный вариант авторизации, но только он позволяет сервису однозначно установить приложение, обращающееся за авторизацией. Во всех остальных вариантах авторизация происходит полностью на клиенте и по понятным причинам возможна маскировка одного приложения под другое. Это стоит учитывать при внедрении OAuth-аутентификации в API сервисов.

### Авторизация полностью клиентских приложений

Авторизация полностью клиентских приложений, отражена на рисунке 3.2.

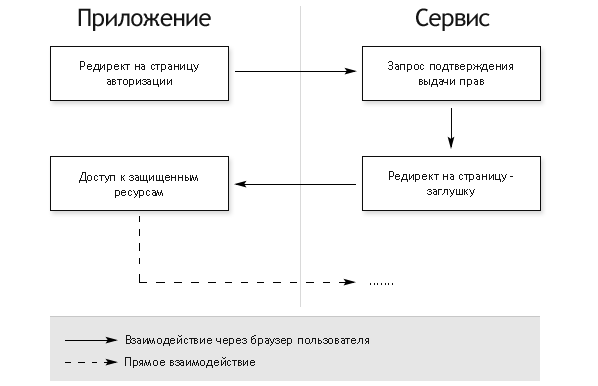


Рисунок 3.2 – Авторизация полностью клиентских приложений

Примерный алгоритм работы:

1. открытие встроенного браузера со страницей авторизации;
2. у пользователя запрашивается подтверждение выдачи прав;
3. в случае согласия пользователя, браузер перенаправляется на страницу-заглушку во фрагменте (после #) URL которой добавляется access token;
4. приложение перехватывает редирект и получает access token из адреса страницы.

Этот вариант требует поднятия в приложении окна браузера, но не требует серверной части и дополнительного вызова сервер-сервер для обмена authorization code на access token.

## Описание проблем при реализации облачного сохранения

В ходе разработки API выбранных сервисов выявлено ряд проблем:

1. Яндекс Диск – авторизация прошла успешно, но блокируется любой доступ к диску клиента, c учетом выдачи приложению прав путем получения токена доступа;
2. Dropbox – малый объем бесплатного облачного пространства и необходимость иметь отельный сайт для приложения (для Outh реализации необходима redirect url) усложнило бы разработку и было бы малоэффективным решением.

Поскольку в ходе разработки два из трех API выбранных сервисов не показал стабильной работы или не соответствовал необходимым критериям в конечной версии программного средства используется API Google диска который полностью справился с поставленной задачей.

Библиотека Accord.Video.FFMPEG показала ряд недостатков:

* низкое качество записи;
* ориентирование на тип процессоров AMD в следствии чего некорректно работала в ходе разработки на процессоре Intel;
* зависимость от мультимедийного фреймворка FFmpeg.

В результате было решено отказаться от использования вспомогательной библиотеки Accord.Video.FFMPEG и воспользоваться библиотекой CSScreenRecorder.

# Тестирование

## Функциональное тестирование

Тестирование программного обеспечения – проверка соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов, выбранном определенным образом. Цель тестирования – выяснить наличие ошибок.

Принципы тестирования, обеспечивающие приемлемую вероятность обнаружения ошибок, сводятся к следующему:

* описание предполагаемых значений входных данных или результатов должно быть необходимой частью тестового набора;
* следует по возможности избегать тестирования программы ее автором;
* тесты для неправильных и непредусмотренных входных данных следует разрабатывать также тщательно, как для правильных и предусмотренных данных;
* необходимо проверять не только, делает ли программа то, для чего она предназначена, но и не делает ли она того, чего не должна делать;
* нельзя планировать тестирование в предположении, что ошибки не будут обнаружены.

Для проверки правильности работы программы были проведены следующие тесты:

* запуск программы;
* регистрация пользователей;
* авторизация пользователей;
* предпросмотр видео;
* запись видео с предпросмотром;
* запись видео с разрешением области по умолчанию;
* запись видео с произвольным разрешением области;
* создание скриншота с разрешением области по умолчанию;
* создание скриншота с произвольным разрешением области;
* выбор области захвата;
* пауза записи;
* возобновление записи;
* сохранение на облачный сервер;
* сохранение настроек пользователя;

Не все тесты были пройдены успешно, что означает, что не все функции выполняются верно. Результаты тестов представлены в таблице 4.1.

Таблица 4.1 – Результаты тестирования компонентов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Входные данные | Ожидаемый результат | Фактический результат | Результат тестирования | Примечание |
| 1 | 2 | 3 | 4 | 5 |
| Выбрать область. | Переменные области захвата изменят значение. | Переменные области захвата изменили значение. | Успешно |  |
| Авторизация пользователя | Пользователь авторизован | Пользователь авторизован | Успешно |  |
| Запись видео с разрешением области по умолчанию | Видео записано. | Видео записано. | Успешно |  |
| Запись видео с произвольным разрешением области. | Видео записано. | Зависание приложения | Неудачно | Библиотека записи видео не может компилировать видео с нечетным разрешением |
| Создание скриншота с разрешением области по умолчанию. | Скриншот создан. | Скриншот создан. | Успешно |  |
| Создание скриншота с произвольным разрешением области. | Скриншот создан. | Скриншот создан. | Успешно |  |
| Пауза записи. | Запись приостановлена. | Запись приостановлена. | Успешно |  |
| Возобновление записи. | Запись возобновлена. | Запись возобновлена. | Успешно |  |
| предпросмотр видео | предпросмотр отображается корректно. | предпросмотр отображается корректно. | Успешно |  |
| запись видео с предпросмотром | Видео записано. | Зависание приложения | Неудачно | Возможно влияет поток конвертации bitmap для предпросмотра |
| сохранение на облачный сервер | Данные сохранены | Данные сохранены | Успешно |  |
| сохранение настроек пользователя | Данные сохранены | Данные сохранены | Успешно |  |

Для решения двух вышеописанных проблем найдены соответствующие им решения:

* корректировка координат с нечетных на четные;
* отказ от функции предпросмотра.

## Тестирование с помощью тест-кейсов

Тест-кейс – набор входных значений, предусловий выполнения, ожидаемых результатов и постусловий выполнения, разработанный для определённой цели или тестового условия, таких как выполнения определённого пути программы или же для проверки соответствия определённому требованию.

Ниже будут представлены проведенные тесты в виде тест-кейсов.

Таблица 4.2 – Тест-кейс №1

|  |  |  |
| --- | --- | --- |
| Заголовок тест-кейса №1 | | |
| Авторизация пользователя в приложении | | |
| Предусловие | | |
| Пользователь не был авторизован в браузере. Открыто окно авторизации. | | |
| № | ШАГ | ОЖИДАЕМЫЙ РЕЗУЛЬТАТ |
|  | Нажать на кнопку "Вход". | Откроется окно браузера для выбора пользователя Google |
|  | Ввести в поле "Телефон или адрес электронной почты" значение состоящее из латинских букв, кириллицы, спецсимволов и чисел. | В поле "Пароль" отображается введённый текст, скрытый символами пароля. Значения корректно считаются в самом приложении. |
|  | Ввести в поле "Введите пароль" значение состоящее из латинских букв, кириллицы, спецсимволов и чисел. | В поле "Пароль" отображается введённый текст, скрытый символами пароля. Значения корректно считаются в самом приложении. |
|  | Нажать кнопку «Далее» | Откроется страница для предоставления доступа к Google диску |
|  | Проверить наличие выделенного пункта «Просмотр создание и удаление всех файлов на Google диске». | Появится сообщении об успехе регистрации. |
|  | Нажать на кнопку "Разрешить". | Появится сообщение об успехе авторизации |
|  | Проверить наличие файла токена доступа. | Файл найден |
| Результат тестирования: | | Тест успешно пройден. |

Таблица 4.3 – Тест-кейс №2

|  |  |  |
| --- | --- | --- |
| Заголовок тест-кейса №2 | | |
| Создание скриншота с произвольным разрешением и сохранение на физический носитель | | |
| Предусловие | | |
| Область была выбрана. | | |
| № | ШАГ | ОЖИДАЕМЫЙ РЕЗУЛЬТАТ |
|  | Нажать на кнопку «Выбрать область». | Появится инструмент выделения области экрана. |
|  | Выбрать область. | В поле «Log» отразится разрешение текущей области. |
|  | Нажать на кнопку «Скриншот». | Откроется диалоговое окно для сохранения. |
|  | Нажать на кнопку «Сохранить на диск». | Откроется стандартное диалоговое окно Windows для выбора папки сохранения. |

Продолжение таблицы 4.3 – Тест-кейс №2

|  |  |  |
| --- | --- | --- |
|  | Выбрать папку. | Закрытие окна сохранения. |
|  | Проверить наличие файла в папке. | Файл найден. |
| Результат тестирования: | | Тест успешно пройден. |

Таблица 4.4 - Тест-кейс №3

|  |  |  |
| --- | --- | --- |
| Заголовок тест-кейса №3 | | |
| Создание скриншота с произвольным разрешением и сохранение в облачный сервер. | | |
| Предусловие | | |
| Область была выбрана. | | |
| № | ШАГ | ОЖИДАЕМЫЙ РЕЗУЛЬТАТ |
|  | Нажать на кнопку «Выбрать область». | Появится инструмент выделения области экрана. |
|  | Выбрать область. | В поле «Log» отразится разрешение текущей области. |
|  | Нажать на кнопку «Скриншот». | Откроется диалоговое окно для сохранения. |
|  | Нажать на кнопку «Сохранить в облаке». |  |
|  | Проверить наличие файла в облачном сервере. |  |
| Результат тестирования: | | Тест успешно пройден. |

Таблица 4.5 – Тест-кейс №4

|  |  |  |
| --- | --- | --- |
| Заголовок тест-кейса №4 | | |
| Запись экрана ПК с произвольным разрешением и сохранение на физический носитель | | |
| Предусловие. | | |
| Область была выбрана. Запись не начата. | | |
| № | ШАГ | ОЖИДАЕМЫЙ РЕЗУЛЬТАТ |
|  | Нажать на кнопку «Выбрать область». | Появится инструмент выделения области экрана. |
|  | Выбрать область. | В поле «Log» отразится разрешение текущей области. |
|  | Нажать на кнопку «Начать запись». | В поле «Log» отразится сообщение о начале записи. |
|  | Нажать на кнопку «Завершить». | В поле «Log» отразится сообщение о начале записи. Откроется диалоговое окно для сохранения. |
|  | Нажать на кнопку «Сохранить на диск». | Откроется стандартное диалоговое окно Windows для выбора папки сохранения. |
|  | Выбрать папку. | Закрытие окна сохранения. |
|  | Проверить наличие файла в папке. | Файл найден. |
| Результат тестирования: | | Тест успешно пройден |

Таблица 4.6 – Тест-кейс №5

|  |  |  |
| --- | --- | --- |
| Заголовок тест-кейса №5 | | |
| Запись экрана ПК с произвольным разрешением и сохранение в облачный сервер. | | |
| Предусловие. | | |
| Область была выбрана. Запись не начата. | | |
| № | ШАГ | ОЖИДАЕМЫЙ РЕЗУЛЬТАТ |
|  | Нажать на кнопку «Выбрать область». | Появится инструмент выделения области экрана. |
|  | Выбрать область. | В поле «Log» отразится разрешение текущей области. |
|  | Нажать на кнопку «Начать запись». | В поле «Log» отразится сообщение о начале записи. |
|  | Нажать на кнопку «Завершить». | В поле «Log» отразится сообщение о начале записи. Откроется диалоговое окно для сохранения. |
|  | Нажать на кнопку «Сохранить в облаке». | Откроется стандартное диалоговое окно Windows для выбора папки сохранения. |
|  | Проверить наличие файла на сервере. | Файл найден. |
| Результат тестирования: | | Тест успешно пройден. |

Таблица 4.7 – Тест-кейс №6

|  |  |  |
| --- | --- | --- |
| Заголовок тест-кейса №6 | | |
| Пауза записи экрана ПК | | |
| Предусловие. | | |
| Запись начата. Запись не приостановлена. | | |
| № | ШАГ | ОЖИДАЕМЫЙ РЕЗУЛЬТАТ |
|  | Нажать на кнопку «Начать запись». | В поле «Log» отразится сообщение о начале записи. |
|  | Нажать на кнопку «Пауза». | В поле «Log» отразится сообщение о приостановке записи. |
| Результат тестирования: | | Тест успешно пройден. |

Таблица 4.8 – Тест-кейс №7

|  |  |  |
| --- | --- | --- |
| Заголовок тест-кейса №7 | | |
| Возобновление записи экрана ПК | | |
| Предусловие. | | |
| Запись начата. Запись приостановлена. | | |
| № | ШАГ | ОЖИДАЕМЫЙ РЕЗУЛЬТАТ |
|  | Нажать на кнопку «Начать запись». | В поле «Log» отразится сообщение о Возобновлении записи. |
|  | Нажать на кнопку «Пауза». | В поле «Log» отразится сообщение о приостановке записи. |
| Результат тестирования: | | Тест успешно пройден. |

Таблица 4.9 – Тест-кейс №8

|  |  |  |
| --- | --- | --- |
| Заголовок тест-кейса №8 | | |
| Запрет доступа приложению к Google диску | | |
| Доступ предоставлен. Авторизация не произведена. | | |
| Запись начата. Запись приостановлена. | | |
| № | ШАГ | ОЖИДАЕМЫЙ РЕЗУЛЬТАТ |
|  | Нажать на кнопку с изображение человека выходящего в дверь. | В поле «Log» отразится сообщение о запрете доступа. |
|  | Проверить что токен доступа удален | Токен доступа удален |
| Результат тестирования: | | Тест успешно пройден. |

Все тест-кейсы были успешно пройдены. Ошибки не обнаружены.

# Экономиечский раздел

## Исходные данные

Исходная информация для расчета экономической части получена в «Фабрика инноваций и решений» и отображена в таблице 5.1.

Таблица 5.1 – Исходная информация

|  |  |  |  |
| --- | --- | --- | --- |
| Показатель | Условное обозначение | Единицы измерения | Значение, рублей |
| Стоимость компьютера | К | Рублей | 2350 |
| Мощность, потребляемая компьютером | МК | КВт/час | 0.481 |
| Полезный фонд времени работы компьютера за год | FK | Час | 2697 |
| Стоимость принтера | П | Рублей | 329 |
| Мощность, потребляемая принтером | МП | КВт/час | 0.24 |
| Полезный фонд времени работы принтера за год | FП | Час | 747 |
| Стоимость 1 кВт/час (с учетом НДС) | С | Рублей | 0.2543 |
| Норма амортизации для оборудования годовая | Наоб | Проценты | 21 |
| Норма амортизации для зданий годовая | Назд | Проценты | 1 |
| Часовая тарифная ставка 13 разряда | ЧТС13 | Рублей | 1.8062 |

Руководитель дипломного проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

## Определение себестоимости программного обеспечения (ПО), как базы для формирования цены

В современных рыночных экономических условиях ПО выступает преимущественно в виде продукции научно-технических организаций, представляющей собой функционально завершенное и имеющее товарный вид ПО, реализуемое покупателям по рыночным отпускным ценам, установленным в результате анализа рыночных условий и переговоров с заказчиком и возможностей тиражирования. Все завершенные разработки ПО являются научно-технической продукцией.

Широкое применение вычислительной техники требует постоянного обновления и совершенствования ПО. Выбор эффективных проектов ПО связан с их экономической оценкой и расчетом экономического эффекта, который может определяться как у разработчика, так и у пользователя.

У разработчика экономический эффект выступает в виде чистой прибыли, остающейся в распоряжении предприятия от реализации ПО, а у пользователя – в виде экономии трудовых, материальных и финансовых ресурсов, получаемой за счет:

* снижения трудоемкости расчетов и алгоритмизации программирования и отладки программ, за счет использования ПО в процессе разработки автоматизированных систем обработки данных;
* сокращения расходов на оплату машинного времени и других ресурсов на отладку программ;
* снижения расходов на материалы;
* ускорение ввода в эксплуатацию новых систем;
* улучшения показателей основной деятельности предприятий в результате использования ПО.

Стоимостная оценка ПО у разработчиков предполагает составление сметы затрат, которая включает следующие статьи:

* затраты на материалы;
* стоимость машинного времени;
* основная заработная плата;
* дополнительная заработная плата;
* отчисление в фонд социальной защиты населения;
* обязательное страхование от несчастных случаев на производстве и профессиональных заболеваний;
* прочие затраты;
* расходы на сопровождение и адаптацию;

На основании сметы затрат рассчитывается себестоимость ПО. Расчет ведется в следующей последовательности:

## Расчет затрат на материалы

По статье «Материалы» отражаются расходы на магнитные носители, бумагу, красящие ленты и другие материалы, необходимые для разработки ПО. Расчет затрат на материалы (М), необходимые для разработки программного обеспечения (ПО) осуществляется по нормативу (Нр) на 100 команд (принимаем равным 1.9 руб.), с учетом общего объема команд (VкПО) (берется учащимся по факту, исходя из полученного им количества команд при разработке ПО).

Определение затрат на материалы осуществляется по формуле (5.1), в рублях:

|  |  |  |
| --- | --- | --- |
|  |  | (5.1) |

Расчет по полученным показателям

## Обоснование трудоемкости разработки ПО

Обоснование трудоемкости разработки ПО ведется следующим образом: на основании опытно-статистического (суммарного) метода, в целом на все разработку, исходя из того, что ее выполняет один программист 2-ой категории. Данные о трудоемкости разработки, определенные опытно-статистическим путём подтверждаю, трудоемкость - 218

Общий объём программного обеспечения – количество команд (VkПО) – 104 команды

Руководитель дипломного проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

### Расчет стоимости машинного времени.

Определение стоимости потребляемой электроэнергии осуществляется по формуле (5.2), в рублях:

|  |  |  |
| --- | --- | --- |
|  | , | (5.2) |

где Квр - коэффициент, учитывающий использование повремени (Квр=0,8)

Кс - коэффициент, учитывающий потери в сети (Kc=1,05)

Расчет по полученным показателям:

### Определение суммы амортизационных отчислений осуществляется по формуле (5.3), в рублях:

|  |  |  |
| --- | --- | --- |
|  | . | (5.3) |

Расчет по полученным показателям

Результаты расчетов сводим в таблицу 5.2.

Таблица 5.2 – Определение стоимости электроэнергии и амортизационных отчислений

|  |  |  |  |
| --- | --- | --- | --- |
|  | Условное | Значение, рублей | |
| Показатель | обозначение | На год (ЗМг) | На 1 машино-час (СМч) |
| Стоимость потребляемой электроэнергии | Сэл | 315.41 | 0.13 |
| Сумма ежегодных амортизационных отчислений | Сао | 562.59 | 0.2 |
| Итого: | ЗМг | 878 | 0.33 |

Расчет затрат на 1 машино-час ведется исходя из стоимости машинного времени на год (графа 3 таблицы 5.1) деленный на полезный фонд времени работы компьютера (FK.) (из таблицы «Исходная информация»)

### Определение затрат на 1 машино-час осуществляется по формуле (5.4), в рублях:

|  |  |  |
| --- | --- | --- |
|  | . | (5.4) |

Расчет по полученным показателям:

### Определение стоимости машинного времени осуществляется по формуле (5.5), в рублях:

|  |  |  |
| --- | --- | --- |
|  | , | (5.5) |

где Тм - машинное время работы в расчёте на программу, час. (70% от трудоемкости разработки программного обеспечения – п.1.2.)

СМч - стоимость одного машино-часа, руб. (итог графы 4 таблицы 5.1)

Расчет по полученным показателям

## Расчет затрат на оплату труда

Расчёт основной заработной платы ведётся исходя из трудоемкости программного обеспечения.

Определение прямой заработной платы осуществляется по формуле (5.6), в рублях:

|  |  |  |
| --- | --- | --- |
|  | , | (5.6) |

где Тi – трудоемкость программного обеспечения (из п. 1.2.), ч.

ЧТС13 – часовая тарифная ставка 13 разряда, руб.

Расчет по полученным показателям:

Премия определяется как 30 % от основной заработной платы.

Определение премий осуществляется по формуле (5.7), в рублях:

|  |  |  |
| --- | --- | --- |
|  |  | (5.7) |

Дополнительная заработная плата (Зд) включает выплаты, предусмотренные законодательством о труде (оплата отпусков, льготных часов, времени выполнения государственных обязанностей и других выплат, не связанных с основной деятельностью исполнителей) и определяется по нормативу в процентах к основной заработной плате.

Расчет по полученным показателям:

Определение дополнительной заработной платы осуществляется по формуле (5.8), в рублях:

|  |  |  |
| --- | --- | --- |
|  |  | (5.8) |

где Зос – основная заработная плата; ЗД – дополнительная заработная плата на конкретное ПО в руб.

НД – норматив дополнительной заработной платы принимается 10% от основной заработной платы.

Расчет по полученным показателям:

Отчисления в фонд социальной защиты населения (3C3) определяются в соответствии с действующими законодательными актами по нормативу в процентном отношении к фонду основной и дополнительной зарплаты (Зд).

Определение отчислений в фонд социальной защиты населения осуществляется по формуле (5.9), в рублях:

|  |  |  |
| --- | --- | --- |
|  |  | (5.9) |

где Нсз– норматив отчислений в фонд социальной защиты населения (процент действующий норматив на момент расчета).

Обязательное страхование от несчастных случаев на производстве и профессиональных заболеваний (Зстр) берётся по нормативу, установленному в целом по организации (можно принять Нстр– 0,6 %).

Расчет по полученным показателям:

Определение отчислений на обязательное страхование от несчастных случаев на производстве и профессиональных заболеваний осуществляется по формуле (5.10), в рублях:

|  |  |  |
| --- | --- | --- |
|  |  | (5.10) |

Расчет по полученным показателям

## Расчет полной себестоимости программного обеспечения

Прочие затраты (Пз) включают расходы, связанные с необходимостью содержания аппарата управления, организационные расходы и расходы на общехозяйственные нужды. Принимаются (Нпз) в процентном отношении к основной заработной плате исполнителей. Норматив устанавливается в целом по организации (Нпз принимаем 99%).

Определение прочих затрат осуществляется по формуле (5.11), в рублях:

|  |  |  |
| --- | --- | --- |
|  |  | (5.11) |

Расчет по полученным показателям

Полная себестоимость без учета расходов на сопровождение и адаптацию (Сп) рассчитывается как сумма затрат на материалы, стоимости машинного времени, затрат на основную, дополнительную заработную плату, стоимости отчислений в фонд социальной защиты населения и на обязательное страхование от несчастных случаев на производстве и профессиональных заболеваний, прочих затрат.

Определение полной себестоимости осуществляется по формуле (5.12), в рублях:

|  |  |  |
| --- | --- | --- |
|  |  | (5.12) |

Кроме того, организация-разработчик осуществляет затраты на сопровождение и адаптацию ПО (РСА), которые определяются по нормативу НРСА (принимаем 4%), от полной себестоимости без учёта расходов на адаптацию (Сп).

Расчет по полученным показателям:

Определение затрат на сопровождение и адаптацию ПО осуществляется по формуле (5.13), в рублях:

|  |  |  |
| --- | --- | --- |
|  |  | (5.13) |

где Сп – полная себестоимость ПО без учёта расходов на сопровождение и адаптацию, руб.

Расчет по полученным показателям:

Определение общей суммы расходов на разработку (с затратами на сопровождение и адаптацию) (Спл) осуществляется по формуле (5.14), в рублях:

|  |  |  |
| --- | --- | --- |
|  |  | (5.14) |

Расчет по полученным показателям

Результаты всех расчётов сводим в таблицу 5.3.

Таблица 5.3 – Определение полной себестоимости программного обеспечения

|  |  |  |
| --- | --- | --- |
| Наименование статей затрат | Сумма, руб. | Обоснование расчета |
| 1. Затраты на материалы | 1.98 | пункт 1.1. |
| 2. Стоимость машинного времени | 50.36 | пункт 1.3. |
| 3. Основная заработная плата | 511.88 | пункт 1.4 |
| 4.Дополнительная заработная плата | 51.19 | 10% от ст.3 (пункт 1.4) |
| 5. Итого, фонд оплаты труда (ФОТ) | 563.07 | (ст.3 + ст.4) (пункт 1.4) |
| 6. Отчисление в фонд социальной защиты населения | 191.44 | 34% от ст.5 (пункт 1.4) |
| 7. Обязательное страхование от несчастных случаев на производстве и профессиональных заболеваний | 3.38 | 0,6 % от ст.5  (пункт 1.4) |
| 8. Итого: отчисления на социальные нужды | 194.82 | (ст. 6 + ст.7) (пункт 1.4) |
| 9. Прочие затраты | 506.76 | 99% от ст.3 (пункт 1.5) |
| 10. Полная себестоимость без учёта расходов на сопровождение и адаптацию | 1316.99 | (ст.1+ст.2+ст.5+ст.8  +ст.9) таблицы 3 (пункт 1.5) |
| 11. Расходы на сопровождение и адаптацию. | 52.68 | 4% от ст.10 таблицы 3 (пункт 1.5) |
| 12. Полная себестоимость с учётом расходов на сопровождение и адаптацию | 1369.67 | (ст.10 + ст.11) таблицы 3 (пункт 1.5) |

Список использованных источников

1. Сайт Metanit, статья C# WPF [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/wpf/.
2. Ермолович, Л.Л. Практикум по анализу хозяйственной деятельности предприятия / Л.Л. Ермолович – Минск: Книжный Дом, 2003.
3. Карпей Т.В. Экономика, организация и планирование промышленного производства / Т.В Карпей – Мн.: Дизайн ПРО, 2004.
4. Dropbox API [Электронный ресурс]. – Режим доступа: https://dropbox.github.io/dropbox-sdk-dotnet/html/R\_Project\_DotNetApiDocumentation.htm
5. Сайт Google Drive API .NET Quickstart [Электронный ресурс]. – Режим доступа: https://developers.google.com/drive/api/v3/quickstart/dotnet?hl=ru.
6. Сайт библиотеки CSScreenRecorder [Электронный ресурс]. – Режим доступа: https://www.microncode.com/developers/cs-screen-recorder/.
7. Quickstart: Azure Blob Storage client library v12 for .NET [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/en-us/azure/storage/blobs/storage-quickstart-blobs-dotnet - Microsoft Ignite, 07.24.2020.
8. Программы для записи видео с экрана [Электронный ресурс]. – Режим доступа: https://softcatalog.info/ru/obzor/programmy-dlya-zapisi-s-ekrana
9. API Яндекс.Диска [Электронный ресурс]. – Режим доступа: https://yandex.ru/dev/disk/api/concepts/about.html.

Заключение

В результате разработки была создано программное средство для захвата с экранов ПК имеющая ряд функций:

* захват видео с экрана ПК в формате AVI;
* настройка качества видео;
* выбор области захвата;
* русскоязычный интерфейс;
* подключение облачного хранилища;

Приложение протестировано и полностью работоспособно, также оно отвечает всем поставленным задачам и реализует ключевую цель программного решения.

Также пришлось столкнутся с радом проблем, описанных ранее в подразделе 3.4. Для их решения была потрачена большая часть времени для разработки программного средства. Были найдены решения или альтернативы средств и программных инструментов в ходе использования которых возникали ошибки или их работа не соответствовала постеленным задачам.

В качестве ключевого сервиса выбран API Google диска так как он хорошо показал себя в работе над поставленными задачами. В качестве основного средства для выполнения главных функций приложения использована библиотека CSScreenRecorder так как данная библиотека показала качественный результат.

Для развития приложения можно добавить дополнительный функционал приложения. Приоритетными функциями будут является:

* добавление новых облачных сервисов;
* создания сайта приложения;
* обновление функций захвата;
* добавление новых форматов файлов;
* подключение популярных социальных платформ таких как YouTube и Twitch.

В ходе дипломного проектирования получены практические и теоретические знания и навыки, которые были успешно применены для разработки дипломного проекта.

Приложение 1

ИСХОДНЫЙ ТЕКСТ ПРОГРАММНОГО СРЕДСТВА

Листинг П1.1 – Класс, описывающий работу окна авторизации

public partial class LoginWindow : Window

{

public LoginWindow()

{

InitializeComponent();

}

private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

{

if (e.LeftButton == MouseButtonState.Pressed)

this.DragMove();

}

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

{

Close();

}

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

{

try

{

DriveAgent agent = new DriveAgent();

Settings.Default.IsLogged = true;

Close();

}

catch (Exception)

{

Settings.Default.IsLogged = false;

}

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

if (System.IO.File.Exists(AppDomain.CurrentDomain.BaseDirectory + ".credentials/drive-dotnet-quickstart.json/Google.Apis.Auth.OAuth2.Responses.TokenResponse-user"))

{

Settings.Default.IsLogged = true;

MessageBox.Show("Вы авторизованы");

Close();

}

else

{

Settings.Default.IsLogged = false;

}

}

Листинг П1.2 – Класс, описывающий работу главного окна приложения

public partial class MainWindow : Window

{

public int AreaHeight { get; private set; }

public int AreaWidth { get; private set; }

public int AreaTop { get; private set; }

public int AreaLeft { get; private set; }

public MainWindow()

{

Settings.Default.IsLogged = false;

InitializeComponent();

}

CSScreenRecorder.ScreenRecorder ScreenRecorder;

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

{

using (ScreenRecorder = new CSScreenRecorder.ScreenRecorder("UserName", "RegKey"))

{

#region Area

ScreenRecorder.RecordScreenHeight = AreaHeight;

ScreenRecorder.RecordScreenWidth = AreaWidth;

ScreenRecorder.RecordScreenTop = AreaTop;

ScreenRecorder.RecordScreenLeft = AreaLeft;

#endregion

#region Video

ScreenRecorder.VideoFPS = 24;

ScreenRecorder.VideoEncoderQuality = Settings.Default.VideoEncoderQuality; //%

ScreenRecorder.VideoCodecIndex = 1; //jpeg commpression

#endregion Video

#region Audio

ScreenRecorder.AudioRecorder = false;

#endregion Audio

ScreenRecorder.TrackMouse = true;

#region Events

ScreenRecorder.OnStartRecord += (o) =>

{

LogText.Text = LogText.Text + DateTime.Now + ": Начало записи\r\n";

};

ScreenRecorder.OnPauseRecord += (o) =>

{

LogText.Text = LogText.Text + DateTime.Now + ": Запись приотсановлена\r\n";

};

ScreenRecorder.OnUnPauseRecord += (o) =>

{

LogText.Text = LogText.Text + DateTime.Now + ": Запись возобновлена\r\n";

};

ScreenRecorder.OnStopRecord += (o) =>

{

LogText.Text = LogText.Text + DateTime.Now + ": Запись завершена\r\n";

};

ScreenRecorder.OnPreview += (o, bmp) =>

{

//nop

};

ScreenRecorder.OnError += (o, ErrorMessage, ErrorNumber) =>

{

MessageBox.Show(ErrorMessage + " (" + ErrorNumber + ")", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

LogText.Text = LogText.Text + DateTime.Now + ": Ошибка - " + ErrorMessage + " (" + ErrorNumber + ")\r\n";

};

#endregion

ScreenRecorder.FileName = $"{Settings.Default.Directory}\\{DateTime.Now:yyyy-MM-dd-HH-mm-ss}.avi"; ;

Settings.Default.Filename = ScreenRecorder.FileName;

ScreenRecorder.Record();

}

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

LoginWindow window = new LoginWindow();

window.Show();

ClaculateResolutionWithDPI(SystemParameters.VirtualScreenWidth, SystemParameters.VirtualScreenHeight, 0, 0);

LogText.Text = DateTime.Now + $": Текущее разрешение области захвата = {AreaWidth}x{AreaHeight}" + Environment.NewLine;

}

private void ClaculateResolutionWithDPI(double width, double heigth, double top, double left)

{

var dpi = System.Windows.Media.VisualTreeHelper.GetDpi(new System.Windows.Controls.Control());

AreaWidth = (int)(width \* dpi.DpiScaleX);

AreaHeight = (int)(heigth \* dpi.DpiScaleY);

AreaLeft = (int)(top \* dpi.DpiScaleX);

AreaTop = (int)(left \* dpi.DpiScaleY);

}

private void Window\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

Environment.Exit(0);

}

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

{

try

{

ScreenRecorder.Stop();

SaveDialog saveDialog = new SaveDialog();

saveDialog.ShowDialog();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

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

{

ScreenRecorder.Pause();

}

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

{

ScreenRecorder.UnPause();

}

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

{

Screenshot();

}

private void Screenshot()

{

using (Bitmap bmp = new Bitmap(AreaWidth, AreaHeight))

{

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

{

string filename = $"ScreenShot-{DateTime.Now:dd-MM-yyyy-hh-mm-ss}.png";

g.CopyFromScreen(AreaLeft, AreaTop, 0, 0, bmp.Size);

bmp.Save($"{Settings.Default.Directory}\\{filename}", ImageFormat.Png);

Settings.Default.Filename = $"{Settings.Default.Directory}\\{filename}";

SaveDialog dialog = new SaveDialog();

dialog.ShowDialog();

}

}

}

private int CoordinatCorrecting(int coodinat)

{

return coodinat % 2 != 0 ? coodinat + 1 : coodinat;

}

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

{

CSScreenRecorder.SpinningSelection SpinningSelection = new CSScreenRecorder.SpinningSelection();

using (ScreenRecorder = new CSScreenRecorder.ScreenRecorder("UserName", "RegKey"))

{

ScreenRecorder.SpinningToolColor = System.Drawing.Color.Orange;

ScreenRecorder.SpinningToolBorderSize = 3;

ScreenRecorder.SpinningToolStyle = System.Drawing.Drawing2D.DashStyle.Dash;

ScreenRecorder.OnSpinningSelection += (o, Cancel, Top, Left, Width, Height) =>

{

if (Cancel)

{

LogText.Text = LogText.Text + DateTime.Now + ": Выделение отменено\r\n";

}

else

{

int coodrdinat;

coodrdinat = CoordinatCorrecting(Top);

AreaTop = coodrdinat;

coodrdinat = CoordinatCorrecting(Left);

AreaLeft = coodrdinat;

coodrdinat = CoordinatCorrecting(Width);

AreaWidth = coodrdinat;

coodrdinat = CoordinatCorrecting(Height);

AreaHeight = coodrdinat;

LogText.Text += DateTime.Now + $": Текущее разрешение области захвата = {AreaWidth}x{AreaHeight}" + Environment.NewLine;

}

};

SpinningSelection = ScreenRecorder.SpinningTool();

}

}

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

{

SettingWindow window = new SettingWindow();

window.Show();

}

private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

{

if (e.LeftButton == MouseButtonState.Pressed)

this.DragMove();

}

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

{

this.Close();

}

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

{

this.WindowState = WindowState.Minimized;

}

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

{

LoginWindow window = new LoginWindow();

window.Show();

}

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

{

WindowState = WindowState == WindowState.Normal ? WindowState.Maximized : WindowState.Normal;

}

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

{

DriveAgent agent = new DriveAgent();

agent.ClearCredentials();

}

}

Листинг П1.3 – Класс, описывающий работу диалогового окна сохранения

public partial class SaveDialog : Window

{

public SaveDialog()

{

InitializeComponent();

}

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

{

WindowState = WindowState.Minimized;

}

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

{

Close();

}

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

{

var dialog = new CommonOpenFileDialog();

dialog.IsFolderPicker = true;

if (dialog.ShowDialog() == CommonFileDialogResult.Ok)

{

File.Move(Settings.Default.Filename, dialog.FileName + "\\" + System.IO.Path.GetFileName(Settings.Default.Filename));

this.Close();

}

}

private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

{

if (e.LeftButton == MouseButtonState.Pressed)

this.DragMove();

}

private async void SaveInCloud\_Click(object sender, RoutedEventArgs e)

{

if (Settings.Default.IsLogged)

{

DriveAgent agent = new DriveAgent();

agent.ProcessChanged += Agent\_ProcessChanged;

Google.Apis.Drive.v3.Data.File file = await agent.UploadAsync(Settings.Default.Filename);

File.Delete(Settings.Default.Filename);

MessageBox.Show("Файл успешно загружен", "Уведомление", MessageBoxButton.OK, MessageBoxImage.Information);

Close();

}

else

{

MessageBox.Show("вы не вошли");

}

}

private void Agent\_ProcessChanged(object sourse, ProgressEventArgs e)

{

Debug.WriteLine(e.GetValue());

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

System.IO.FileInfo file = new System.IO.FileInfo(Settings.Default.Filename);

SaveProgress.Maximum = file.Length;

SaveProgress.Minimum = 0;

SaveProgress.Value = 0;

}

}

Листинг П1.4 – Класс, описывающий работу окна настроек

public partial class SettingWindow : Window

{

public CSScreenRecorder.ScreenRecorder ScreenRecorder { get; private set; }

public SettingWindow()

{

InitializeComponent();

}

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

{

WindowState = WindowState.Minimized;

}

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

{

WindowState = WindowState == WindowState.Normal ? WindowState.Maximized : WindowState.Normal;

}

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

{

Close();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

using (ScreenRecorder = new CSScreenRecorder.ScreenRecorder("UserName", "RegKey"))

{

#region Load App Properties

//Directory

DirectoryText.Text = Settings.Default.Directory;

//Video quality

PercentSlider.Value = Settings.Default.VideoEncoderQuality;

//Track mouse

TrackMouseMode.IsChecked = Settings.Default.IsTrackMouse;

#endregion

}

}

private void Window\_MouseDown(object sender, MouseButtonEventArgs e)

{

if (e.LeftButton == MouseButtonState.Pressed)

this.DragMove();

}

private void PercentSlider\_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)

{

if (PescentText != null)

PescentText.Text = PercentSlider.Value + "%"; ;

}

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

{

Close();

}

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

{

var dialog = new CommonOpenFileDialog();

dialog.IsFolderPicker = true;

dialog.ShowDialog();

DirectoryText.Text = dialog.FileName;

}

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

{

#region Load App Properties

Settings.Default.VideoEncoderQuality = (int) PercentSlider.Value;

Settings.Default.Directory = DirectoryText.Text;

Settings.Default.IsTrackMouse = TrackMouseMode.IsChecked.Value;

Settings.Default.Save();

#endregion

Close();

}

Листинг П1.5 – Класс, отвечающий за взаимодействие с облачным хранилищем

public delegate void ProgressHandler(object sourse, ProgressEventArgs e);

public class ProgressEventArgs : EventArgs

{

private int EventInfo;

public ProgressEventArgs(int value)

{

EventInfo = value;

}

public int GetValue()

{

return EventInfo;

}

}

public class DriveAgent

{

string[] Scopes = { DriveService.Scope.Drive };

string ApplicationName = "cafeasp demo 01";

public event ProgressHandler ProcessChanged;

DriveService driveService;

UserCredential credential;

public DriveAgent()

{

credential = GetCredentials();

driveService = new DriveService(new BaseClientService.Initializer()

{

HttpClientInitializer = credential,

ApplicationName = ApplicationName,

});

}

//oauth authorization method

private UserCredential GetCredentials()

{

UserCredential credential;

try

{

using (var stream = new FileStream("credentials.json", FileMode.Open, FileAccess.Read))

{

//save token

string credPath = AppDomain.CurrentDomain.BaseDirectory;

credPath = Path.Combine(credPath, ".credentials/drive-dotnet-quickstart.json");

//oauth authorization

credential = GoogleWebAuthorizationBroker.AuthorizeAsync(

GoogleClientSecrets.Load(stream).Secrets,

Scopes,

"user",

CancellationToken.None,

new FileDataStore(credPath, true)).Result;

}

}

catch (Exception)

{

throw;

}

return credential;

}

public void ClearCredentials()

{

try

{

if (System.IO.File.Exists(AppDomain.CurrentDomain.BaseDirectory + ".credentials/drive-dotnet-quickstart.json/Google.Apis.Auth.OAuth2.Responses.TokenResponse-user"))

{

credential.RevokeTokenAsync(CancellationToken.None);

}

}

catch (Exception)

{

throw;

}

}

//resumable upload file method

public async Task<Google.Apis.Drive.v3.Data.File> UploadAsync(string filepath)

{

string destfilename = Path.GetFileName(filepath);

List<string> parents = new List<string>();

parents.Add("root");

// Prepare the JSON metadata to define file

string json = "{\"name\":\"" + destfilename + "\"";

if (parents.Count > 0)

{

json += ", \"parents\": [";

foreach (string parent in parents)

{

json += "\"" + parent + "\", ";

}

json = json.Remove(json.Length - 2) + "]";

}

json += "}";

Debug.WriteLine(json);

//resumable upload file

Google.Apis.Drive.v3.Data.File uploadedFile = null;

try

{

System.IO.FileInfo info = new System.IO.FileInfo(filepath);

ulong fileSize = (ulong)info.Length; //file size in bytes

//start file stream

var uploadStream = new System.IO.FileStream(filepath, System.IO.FileMode.Open, System.IO.FileAccess.Read);

var insert = driveService.Files.Create(new Google.Apis.Drive.v3.Data.File { Name = destfilename, Parents = new List<string> { "root" } }, uploadStream, "application/octet-stream");

Uri uploadUri = insert.InitiateSessionAsync().Result;

int chunk\_size = ResumableUpload.MinimumChunkSize;

int bytesSent = 0;

//start upload session

while (uploadStream.Length != uploadStream.Position)

{

byte[] temp = new byte[chunk\_size];

int cnt = uploadStream.Read(temp, 0, temp.Length);

if (cnt == 0)

break;

HttpWebRequest httpRequest = (HttpWebRequest)WebRequest.Create(uploadUri);

httpRequest.Method = "PUT";

httpRequest.Headers["Authorization"] = "Bearer " + ((UserCredential)driveService.HttpClientInitializer).Token.AccessToken;

httpRequest.ContentLength = (long)cnt;

httpRequest.Headers["Content-Range"] = string.Format("bytes {0}-{1}/{2}", bytesSent, bytesSent + cnt - 1, fileSize);

using (System.IO.Stream requestStream = httpRequest.GetRequestStreamAsync().Result)

{

requestStream.Write(temp, 0, cnt);

}

HttpWebResponse httpResponse;

try

{

httpResponse = (HttpWebResponse)httpRequest.GetResponse();

}

catch (WebException ex)

{

httpResponse = (HttpWebResponse)ex.Response;

}

if (httpResponse.StatusCode == HttpStatusCode.OK)

{ }

else if ((int)httpResponse.StatusCode != 308)

break;

bytesSent += cnt;

ProcessChanged(this, new ProgressEventArgs(bytesSent));

}

if (bytesSent != uploadStream.Length)

{

return null;

}

FilesResource.ListRequest request = driveService.Files.List();

if (parents.Count > 0)

request.Q += "'" + parents[0] + "' in parents and ";

request.Q += "name = '" + destfilename + "'";

FileList result = request.Execute();

if (result.Files.Count > 0)

uploadedFile = result.Files[0];

//stop file stream

uploadStream.Close();

}

catch (Exception ex)

{

Debug.WriteLine(ex.Message);

}

return uploadedFile;

}

}

Приложение 2

БЛОК-СХЕМА АЛГОРИТМА РАБОТЫ ПРОГРАММНОГО СРЕДСТВА

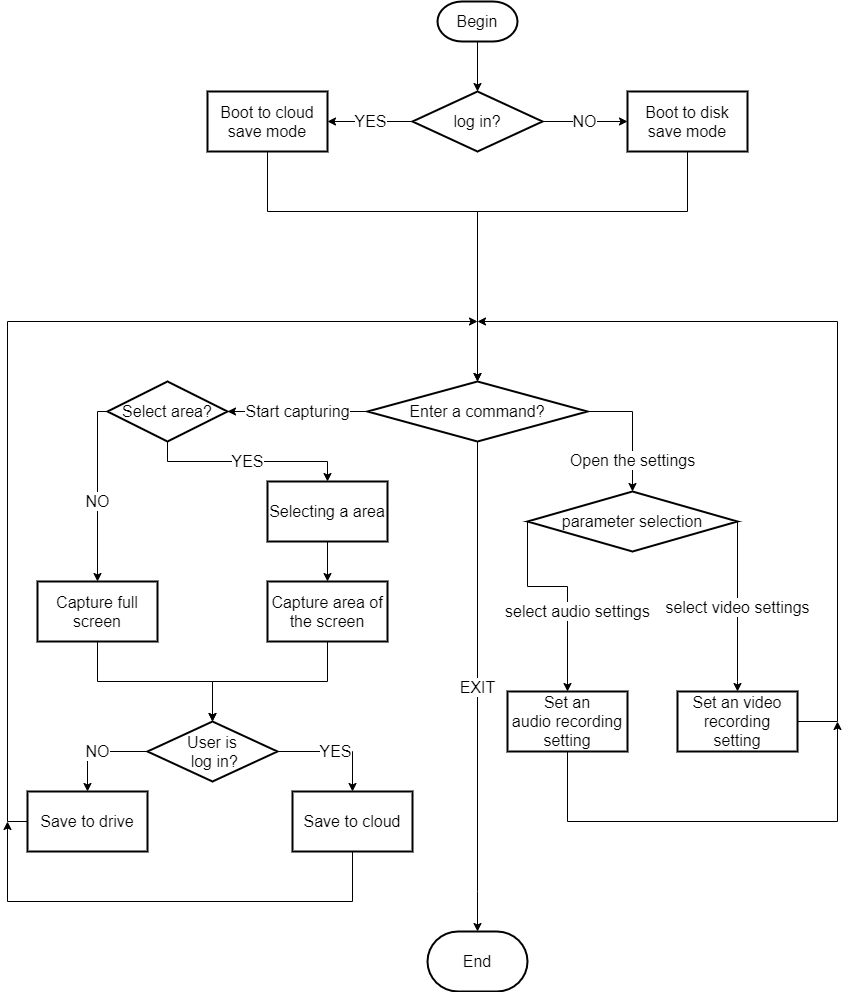


Рисунок П2.1 – блок-схема алгоритма работы программного средства

Приложение 3

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Для регистрации необходимо заполнить поля логин и пароль и нажать на кнопку «Регистрация» (рисунок П3.1).

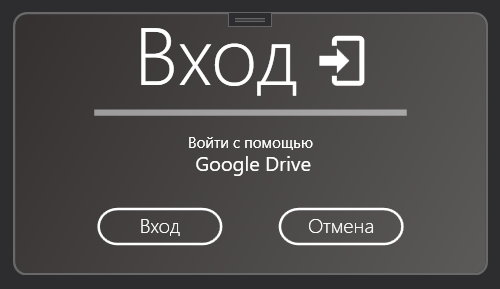


Рисунок П3.1 – Окно входа

Для авторизации на окне входа (рисунок П3.1) необходимо заполнить поля логин и пароль и нажать на кнопку «Вход» после чего отроется окно браузера (рисунок П3.2)

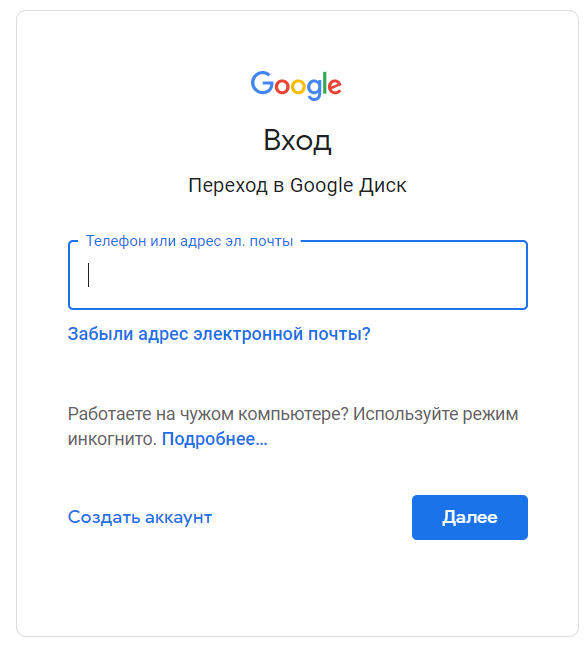


Рисунок П3.2 – Страница входа

Введя свой адрес электронной почты следует нажать на кнопку «Далее». После чего отроется страница пароля и ввести пароль (рисунок П3.3). После ввода пароля нажать на кнопку «Далее»

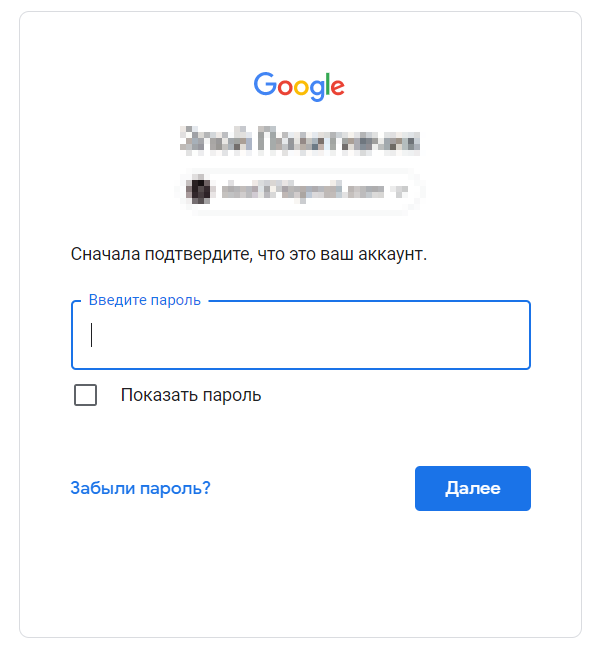


Рисунок П3.3 – Страница ввода пароля

Если пользователь уже авторизован в google аккаунте в браузере по умолчанию, то следует просто выбрать необходимый аккаунт (рисунок П3.4)

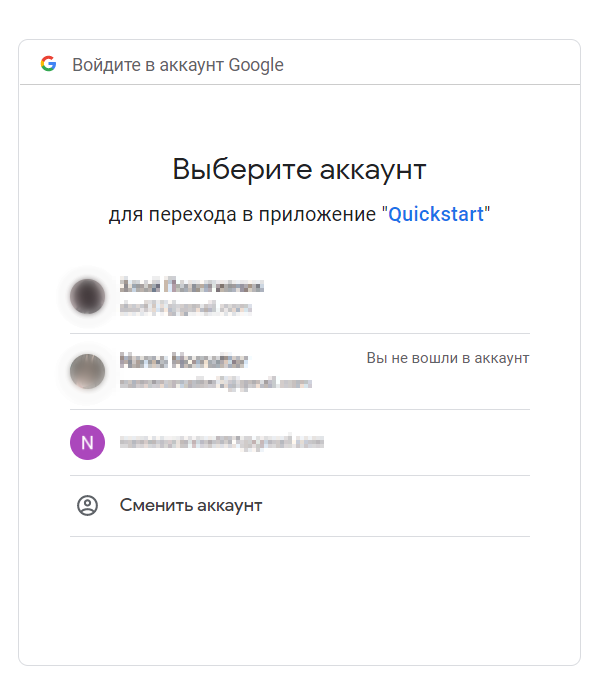


Рисунок П3.4 – Выбор аккаунта

Затем выбрать аккаунт (см. рисунок П3.4). Далее откроется окно для предоставления доступа приложении.

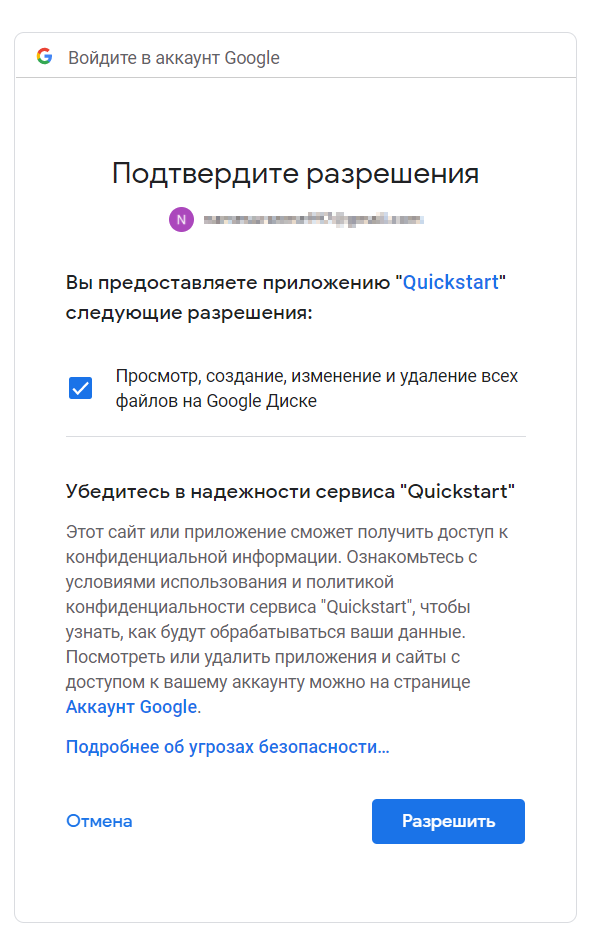


Рисунок П3.5 – Разрешение доступа

После нажатия кнопки «Разрешить» (см рисунок П3.5) на вкладке браузера появится надпись «Received verification code. You may now close this window» далее можно закрыть браузер и авторизация произведена.

На рисунке П3.6 отображено главное окно приложения.

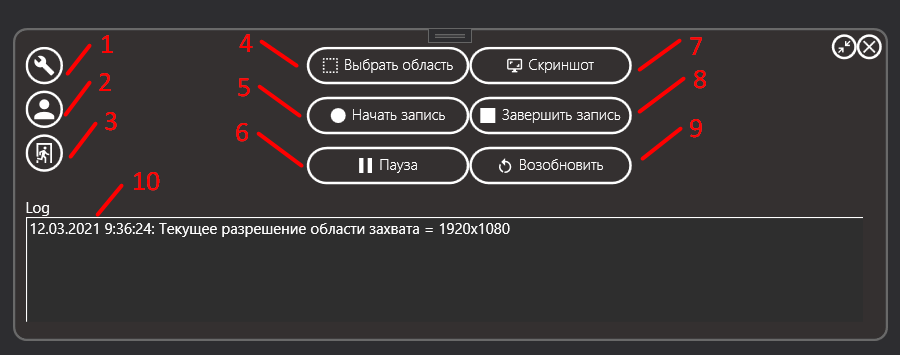


Рисунок П3.6 – Главное окно приложения

Главное окно имеет ряд кнопок:

1. открывает окно настроек;
2. открывает окно входа;
3. закрывает доступ приложения к Google диску;
4. запускает функцию выбора области захвата;
5. начинает запись захвата экрана;
6. приостанавливает запись захват экрана;
7. создает скриншот экрана;
8. завершает запись захвата экрана;
9. возобновляет приостановленную запись захвата экрана;
10. панель отображения лога.

Диалоговое окно сохранения отображено на рисунке П3.7.

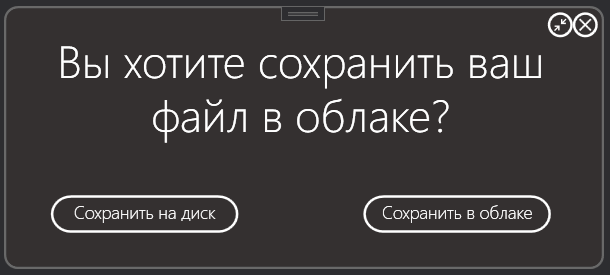


Рисунок П3.7 – Окно сохранения

При нажатии на копку «Сохранить в облако» файл сохраняется в облачном сервере, дальнейших действий не требуется.

При нажатии на копку «Сохранить на диск» откроется стандартное диалоговое окно Windows для выбора папки сохранения, после выбора папки, файл сохраняется.

Окно настроек отображено на рисунке П3.8

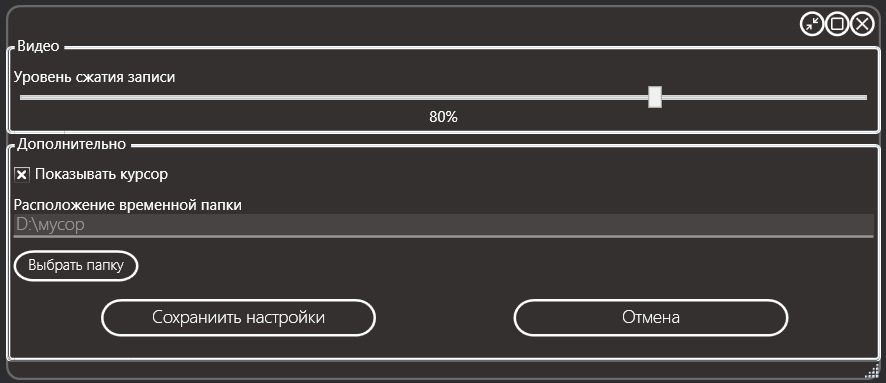


Рисунок П3.8– Окно настроек приложения

Данное окно разделено на две области: настройки видео и дополнительные настройки.

В области дополнительных с настроек можно настроить то будет ли отображаться курсор при записи видео и выбрать папку для хранения временных файлов.

В области видео можно установить степень сжатия видео.

Приложение 4

ПРОТОКОЛ ИСПЫТАНИЙ ПРОГРАММНОГО СРЕДСТВА

Комиссия в составе:

* руководитель дипломного проекта – Разумов С. М.
* преподаватель дисциплин специального цикла – Сутович С.Г
* председатель цикловой комиссии – Тыбербай С. Г.

произвела проверку работоспособности программного средства разработанного в порядке выполнения дипломного проекта.

Таблица П4.1 – Протокол испытаний программного средства

|  |  |
| --- | --- |
| 1. Информация о программном средстве | |
| 1.1. Название программного средства | «Программное средство для захвата изображений и видео с экранов ПК» |
| 1.2. Номер версии программного средства | 1.2.1 |
| 1.3. Дата разработки версии программного средства | 21.03.2021 |
| 1.4. Назначение программного средства | Программное средство предназначено для захвата изображений и видео с экранов ПК |
| 1.5. Функции программного средства | Программное средство позволяет осуществлять захватывать видео и изображения экрана ПК, имеется функция сохранения создаваемых данных в облачное хранилище Google . |
| 1.6. Язык программирования | С# |
| 1.7. Наличие программы инсталляции | Нет в наличии |
| 1.8. Тип компьютера(ов) | ПК, Ноутбук |
| 1.9. Требования к компьютеру  Оперативная память (ОЗУ)  Жесткий диск(и)  Монитор(ы) | 6 Гб и выше  512 Мб и более  Разрешение 1280×720 и выше |
| 1.10 Дополнительное периферийное оборудование: принтеры, мышь, модем и т.п. | Манипулятор типа мышь |
| 1.11 Тип операционной системы (систем) | Windows 7 и выше |
| 2. Информация об организации-заказчике | |
| 2.1 Полное официальное наименование | Витебский государственный политехнический колледж ВГТУ |
| 2.2 Место нахождения (почтовый адрес) | ул. Черняховского, д.14 |
| 2.3 Номер контактного телефона(ов) | (0212) 64-42-97 (директор), 64-89-16 (приемная директора), 64-89-23(бухгалтерия), |
| 2.4 Номер факса | (0212) 64-89-16 |
| 2.5 Адрес электронной почты | vgpt@vitebsk.by |
| 3. Информация о разработчике программного средства | |
| 3.1 Полное официальное наименование | Витебский государственный политехнический колледж ВГТУ |
| 3.2 Место нахождения (почтовый адрес) | 210016 |
| 3.3 Фамилия, инициалы учащегося | Синяк А. И. |
| 3.4 Номер контактного телефона (ов) | (8-0212) 52-86-74 |
| 3.5 Номер факса |  |
| 3.6 Адрес электронной почты | sidusore@mail.ru |

Работоспособность программного средства проверена на контрольных примерах, предусматривающих все возможные варианты исходных и промежуточных данных.

Программное средство признано работоспособным, соответствующим техническому заданию и позволяет решать поставленные перед ним производственные задачи.

Руководитель дипломного проекта Разумов С. М.

Преподаватель дисциплин специального цикла Сутович С.Г

Председатель цикловой комиссии Тыбербай С. Г.