МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Кафедра математического и компьютерного моделирования

РАБОТА ПРОВЕРЕНА
Рецензент, менеджер проекта
ООО «Наполеон АйТи»
_____/ А.А. Завьялов
«____»_____2019 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой МиКМ,
д-р физ.-мат. наук, доцент
_____/ С.А.Загребина
«____»_____2019 г.

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ АВТОМАТИЗАЦИИ РАБОТЫ ФОТОКАБИНЫ

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ–02.03.01.2019.26.ВКР

Руководитель работы, старший
преподаватель кафедры МиКМ
_____/ А.К. Богушов
«____»_____2019 г.

Автор работы,
студент группы ЕТ-411
_____/ С.П. Башков
«____»_____2019 г.

Нормоконтролер,
доцент кафедры МиКМ,
канд. физ.-мат. наук
_____/ А.А. Акимова
«____»_____2019 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Факультет математики, механики и компьютерных технологий
Кафедра математического и компьютерного моделирования
Направление подготовки 02.03.01 Математика и компьютерные науки

УТВЕРЖДАЮ
Заведующий кафедрой,
д.ф.-м.н., доцент

_____/ С.А. Загребина
_____2018 г.

З А Д А Н И Е

на выпускную квалификационную работу студента
Башкова Сергея Павловича,
группа ЕТ-411

1. **Тема работы** «Программное обеспечение автоматизации работы фотокабины»
   утверждена приказом по университету от « 04 » апреля 20 19 г. № 580
2. **Срок сдачи студентом законченной работы** «25» июня 2019 г.
3. **Исходные данные к работе**
   3.1. Мак-Дональд М. WPF: Windows Presentation Foundation в .NET 4.5 с примерами на C# 5.0 для профессионалов. 4-е изд. – М.: Вильямс, 2013. – 1024 с.
   3.2. Троелсен Э. Язык программирования C# 5.0 и платформа .NET 4.5. 6-е изд. – М.: Вильямс, 2015. – 1312 с.
   3.3. Язык программирования C# 5.0.
4. **Перечень вопросов, подлежащих разработке**
   4.1. Выполнить анализ предметной области и произвести обзор существующих решений.
   4.2. Спроектировать архитектуру программного обеспечения.
   4.3. Выполнить реализацию приложения.
   4.4. Провести тестирование.

5. Календарный план

Наименование этапов выпускной квалификационной работы	Срок выполнения этапов работы	Отметка о выполнении руководителя
1) Изучение литературы по тематике выпускной квалификационной работы	01.12.18 – 31.01.19	
2) Обзор методов исследования, применяемых по тематике ВКР	10.01.19 – 28.02.19	
3) Формулировка целей и задач выпускной квалификационной работы	01.03.19 – 18.03.19	
4) Решение основных задач ВКР	11.03.19 – 28.05.19	
5) Разработка структуры выпускной квалификационной работы	22.04.19 – 30.04.19	
6) Подготовка текста выпускной квалификационной работы	29.04.19 – 28.05.19	
7) Проверка работы руководителем, исправление замечаний	03.06.19 – 17.06.19	
8) Нормоконтроль	03.06.19 – 20.06.19	
9) Рецензирование, представление зав. кафедрой	21.06.19 – 27.06.19	
10) Подготовка иллюстративного материала и доклада	17.06.19 – 30.06.19	

6. Дата выдачи задания «01» декабря 2018 г.

Руководитель работы _____/ А.К. Богушов
(подпись)

Студент _____/ С.П. Башков
(подпись)

АННОТАЦИЯ

Башков С.П. Программное обеспечение автоматизации работы фотокабины. Челябинск: ЮУрГУ, ЕТ-411, 32 с., 14 ил., 1 табл., библиогр. список – 20 наим.

Квалификационная работа выполнена с целью разработки программного обеспечения для автоматизации работы фотокабины.

В квалификационной работе были проанализированы готовые решения поставленной задачи, спроектирована архитектура приложения с применением фреймворка .Net Core, разработаны блок схемы функций.

Реализовано программное обеспечение с помощью высокоуровневого языка программирования C# 5.0 и платформы .NET 4.5. Разработан функционал приложения включающий в себя функции инициализации и подключения к фотокамере, установки основных настроек фотокамеры (светочувствительности, диафрагмы, выдержки), цикличного фотографирования пользователя и отправки готового изображения на печать.

По результатам выполненной работы было произведено тестирование программного обеспечения на стабильность в условиях продолжительной непрерывной работы.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы исследования. Индустрия интерактивных зон на мероприятиях различного рода является востребованным и прибыльным бизнесом. Одна из самых популярных интерактивных зон на сегодняшний день является фотокабина.

Фотокабина представляет собой мобильное устройство в виде двух переносных коробов, соединенных между собой и имеющих общий размер 0,5х0,5х1,70 м. Внутрь этой конструкции помещены: печатающее устройство (принтер марки Epson P50), управляющее устройство (ноутбук с установленной операционной системой Windows 10), зеркальный фотоаппарат (Canon EOS 600D), блок взаимодействия программного обеспечения с пользователем (Arduino UNO с подключенными механическим кнопками), встроенное в корпус освещение (светодиодная лента с рассеивателем).

Рассмотрим принцип работы фотокабины. Гость подходит к устройству, надевает развлекательный реквизит (маски, парики, очки и т.д.), делает три или четыре фотографии с интервалом в пять секунд между каждым снимком, после чего спустя 15 секунд получает распечатанный брендированный фотоснимок.

Для функционирования фотокабины необходимо специализированное программное обеспечение. Оно позволяет автоматизировать работу всех модулей фотокабины, а именно: сфотографировать гостя, наложить фотографии на заготовленный заранее брендированный шаблон и отправить получившееся изображение в печать. На рынке программного обеспечения уже есть достаточно много готовых решений, однако их главный минус – это цена. В среднем цена на программное обеспечение для фотокабины варьируется от 25 до 50 тысяч рублей.

Цель и задачи работы. **Целью** данной работы является разработка программного обеспечения для автоматизации работы фотокабины.

Для разработки приложения необходимо решить следующие **задачи**:

1) выполнить анализ предметной области и произвести обзор существующих решений;
2) спроектировать архитектуру программного обеспечения;
3) выполнить реализацию приложения;
4) провести тестирование.

Структура и объем работы. Работа состоит из введения, трёх разделов, заключения, и библиографического списка. Объем работы составляет 32 страницы, объем библиографии – 20 источников, количество иллюстраций – 14.

Во введении обоснована актуальность разработки данного приложения, определены цели и задачи работы, указано краткое содержание, структура и объем работы.

Первая глава «Анализ предметной области» содержит постановку задачи и обзор аналогичных проектов.

Вторая глава «Проектирование приложения» содержит описание и анализ требований к разрабатываемому приложению, а также блоку взаимодействия приложения с пользователем.

Третья глава «Реализация приложения» описывает подробности реализации программного обеспечения и блока для взаимодействия приложения с пользователем. Так же приведены результаты тестирования приложения на устойчивость.

В заключении описываются основные результаты, полученные при выполнении выпускной квалификационной работы.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И СУЩЕСТВУЮЩИХ РЕШЕНИЙ

1.1. Предметная область

Первый фотоавтомат был представлен на Всемирной выставке в Париже в 1889 году. Изобрел это устройство француз по имени Теофил Энгельберт.

Более широко фотокабины начали использоваться в 20-е годы XX века. В 1924 году в Нью-Йорке, на самой оживленной и знаменитой улице – Бродвее начала свою работу фотокабина, созданная американцем с русскими корнями Анатолем Джозефо (Йозефович) [2]. Кабины, делавшие снимок без участия фотографа, и сразу же проявлявшие фотографию, пользовались огромной популярностью во всем мире. Аренда фотобудки в начале XX века могла сделать предпринимателя настоящим богачом, разумеется, при условии, что он мог позволить себе такое вложение. Стоит отметить, что популярность фотокабин была настолько велика, что компания, владевшая ими, была продана в конце 20-х годов за миллион долларов США – по тем временам, целое состояние.

С 1965 по 1979 года фотокабины были запрещены на территории СССР. За ее установку владелец мог получить по меньшей мере административное наказание так как во время правления коммунистической партии все «буржуазное» было строго настрого запрещено, фотокабины в том числе [17]. Но уже в 90-е годы 20 века фотокабины стали активно появляться на людных улицах России, в частности в Москве и Санкт-Петербурге. Посетителями были взрослые и дети. Также в это время можно было выбрать печать цветного фото.

Все фотокабины 20 века были аналоговыми, автоматизированными с помощью применения механических процессов. Устройство состояло из двух отсеков, в одном из которых располагался клиент, а в другом находилось фотооборудование. Так как процесс аналоговой фотографии сильно отличается от современного цифрового фото, отсек с оборудованием содержал погружной фильм-процессор револьверного типа, производивший химико-фотографическую обработку снимков [1]. Из-за этого ранние модели автоматов нуждались в подключении к водопроводу и канализации. Помимо этого, в фотокабинах

находился купюроприёмник, фотоаппарат и осветительные фотовспышки. Съёмка велась без промежуточного негатива, благодаря чему время готовности фотографий не превышало 3-5 минут, фотографии представляли собой фотополоски, чаще всего из трех кадров [20].

В настоящее время популярность набирают цифровые фотокабины, принцип работы которых не многим отличается от фотокабин конца 20 века. Благодаря внедрению технологий, процесс работы фотокабины стал проще и быстрее. Так, в связи с использованием термосублимационных и струйных принтеров, время печати фотографии сократилось с 3 минут до 15 секунд [12]. Специализированное программное обеспечение, в свою очередь, позволило добиться большей функциональности фотокабины.

1.2. Обзор существующих решений

1.2.1. Фотокабина

«Фотокабина» – это программа для фотокабины, разработанная командой lux-foto.com и реализующая процесс автоматизации работы фотокабины. В данной программе осуществлена возможность сделать мгновенную фотографию, а также имеется возможность печати фотографий с карты памяти [11].

К плюсам данной программы можно отнести:

1) присутствие русской локализации приложения;
2) возможность изменить количество фотографий для каждого цикла работы программы;
3) реализована возможность использовать пользовательское изображение шаблона;
4) по окончанию фотографирования реализован выбор наложения художественных фильтров на фотографию.

К минусам можно отнести:

1) отсутствие возможности делать дубликат последней сделанной фотографии;

2) слишком малые размеры окна предпросмотра фотографии;

3) невозможность изменить интервал между каждой фотографией;

4) высокая стоимость лицензии для активации программы.

На рисунке 1.1 представлено изображение интерфейса главного окна программы.

Рисунок 1.1 – Интерфейс главного окна «Фотокабина»

1.2.2. SOCIAL BOOTH

Программа «Social Booth» для фотобудки разработана командой сайта photoboothsolutions.com и работает на операционной системе Windows. Данная программа имеет самый большой функционал по сравнению с остальными конкурентами. Поддерживает зеркальные, цифровые фотокамеры, а также веб-камеры с возможностью сделать фотографии [18]. Имеется возможность записывать видео, делать замедленную съемку (SlowMo) [5], накладывать художественные фильтры по окончанию фотографирования, заменять фон каждой фотографии посредством использования хромакея в качестве фона, делать

анимированные GIF-файлы, печатать и отправлять полученные фото в социальные сети и на электронную почту. Также в комплекте данного программного обеспечения присутствуют предустановленные шаблоны. К минусам данной программы можно отнести отсутствие русской локализации и документации, громоздкость интерфейса, сложность первоначальной настройки программы перед первым использованием. Стоит также отметить высокую стоимость лицензии – 300$ в год. На рисунке 1.2 представлен скриншот окна настроек программы.

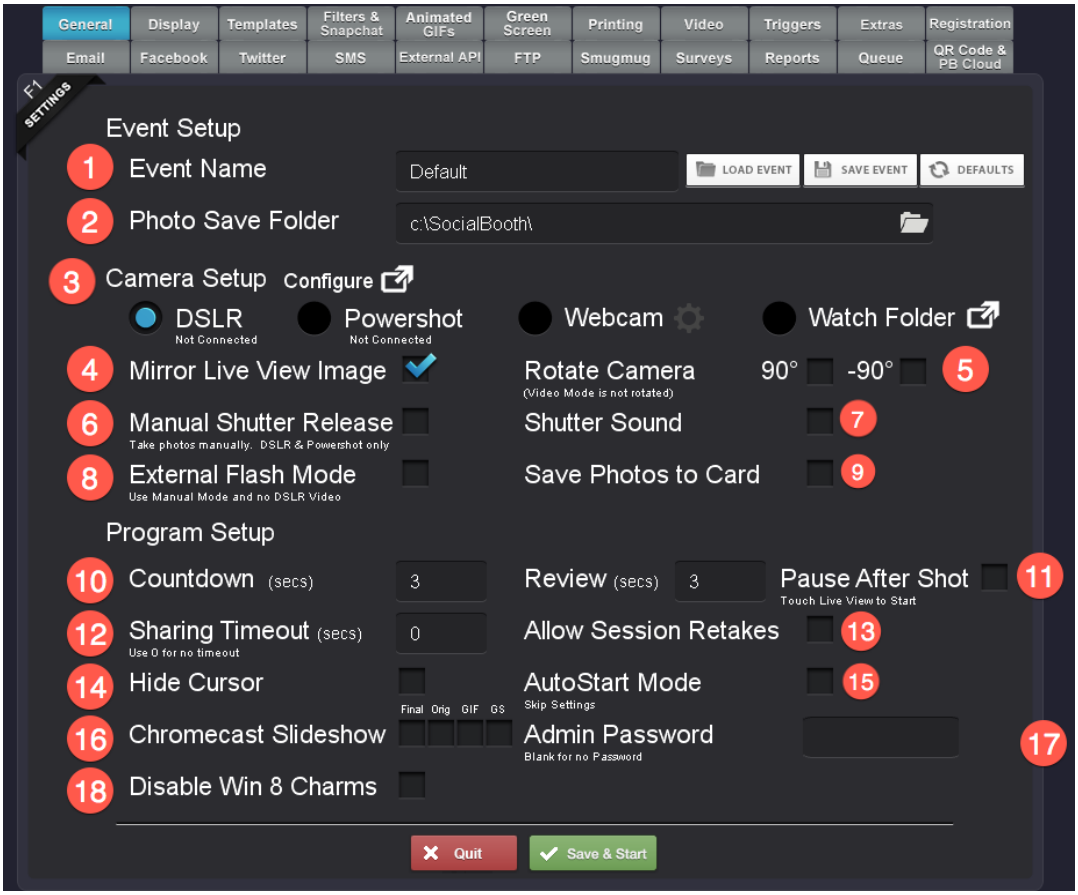

Рисунок 1.2 – Окно настроек программы «Social Booth»

1.2.3. Фотофабрика

Данная программа была разработана командой photofabrika.ru для автоматизации процесса работы фотокабины. Работает на операционной системе Windows. Поддерживает цифровые камеры марки Canon и Nikon. Для клиента доступен выбор один из двух заранее заготовленных шаблонов: 5x15 см и

10x15 см [19]. Далее происходит съемка трех или четырех кадров, в зависимости от выбранного варианта, после чего клиент получает свой брендированный снимок. В программе так же предусмотрены настройки светочувствительности, диафрагмы и выдержки камеры [8]. Реализована функция сохранения всех цифровых копий фотографий в памяти ноутбука. Так же реализована функция предпросмотра изображения с камеры перед началом процесса фотографирования. Положительным стоит отметить наличие вспомогательных надписей которые помогают ориентироваться в программе неподготовленному пользователю. Окно процесса фотографирования представлено на рисунке 1.3. К минусам можно отнести невозможность редактировать время задержки между кадрами и отсутствие функции печати дубликата фотографии.

Рисунок 1.3 – Окно настроек программы «Social Booth»

1.3. Выводы по первой главе

В данном разделе рассмотрено строение фотокабины, особенности программного обеспечения, а также наиболее распространенные функции. Самым

функциональным является приложение «Social Booth», но в силу отсутствия русской локализации и сложности первоначальной настройки данное приложение уступает своим конкурентам. В программе «Фотокабина» отсутствует возможность настройки параметров камеры. Программное обеспечение «Фотофабрика» является оптимальным решением поставленной задачи, так как присутствует весь минимально необходимый функционал, но в связи с высокой стоимостью не рассматривается как решение поставленной задачи. Сравнительную таблицу всех функций каждого программного обеспечения можно ознакомиться в таблице 1.1.

Таблица 1.1 – Сравнение функционала ПО фотокабин

	«Фотокабина»	«Social Booth»	«Фотофабрика»
Поддержка цифровых фотокамер Canon	+	+	+
Изменение светочувствительности камеры	–	+	+
Изменение диафрагмы камеры	–	+	+
Изменение времени выдержки фотокамеры	–	+	+
Предустановленные шаблоны фотографий	+	+	–
Изменение времени задержки между каждой фотографией	–	+	–
Возможность видеозаписи роликов	–	+	–
Русскоязычный интерфейс программы	+	–	+
Печать дубликата последней фотографии	–	–	+
Сохранение электронных копий фотографий на компьютер	+	+	+

2. ПРОЕКТИРОВАНИЕ

2.1. Требования

2.1.1. Функциональные требования

Разрабатываемое приложение должно удовлетворять следующим функциональным требованиям:

1) программа должна поддерживать цифровые камеры Canon;
2) программа должна поддерживать изменение ISO (светочувствительности) фотокамеры;
3) программа должна поддерживать изменение диафрагмы фотокамеры;
4) программа должна поддерживать изменение времени выдержки фотокамеры;
5) в программе должны быть реализованы предустановленные шаблоны для фотографий;
6) в программе должна быть реализована возможность изменить время задержки между каждым кадром;
7) в программе должна быть реализована функция записи видеороликов;
8) в программе должна быть реализована функция печати дубликата последней фотографии;
9) в программе должна быть реализована функция сохранения всех электронных копий фотографий на компьютер.

2.1.2. Нефункциональные требования

Разрабатываемое приложение должно удовлетворять следующим нефункциональным требованиям:

1) разрабатываемое приложение должно быть устойчиво к продолжительной бесперебойной работе;
2) разрабатываемое приложение должно иметь простой и понятный интерфейс.

2.2. Обоснование выбора средств разработки

Для написания программного обеспечения фотокабины нужен мощный и очень гибкий язык программирования. Потому что, в любой момент может потребоваться изменить функционал программы и это необходимо сделать в рекордно короткие сроки.

2.2.1. Язык C#, платформа .Net Core 2.0, технология WPF

На сегодняшний день C# является одним из самых гибких и мощных языков программирования. На нем пишутся совершенно различные программы: от небольших десктопных приложений до крупных веб-сервисов и веб-порталов, которые обслуживают миллионы людей в день. Этот язык является полностью объектно-ориентированным. Поэтому этот подход позволяет решить задачи по построению гибких, но в тоже время сложных, приложений.

Язык C# был специально создан для работы с платформой .Net. И после выхода фреймворка .Net Core, теперь приложения, написанные на нем, можно без проблем запускать не только на операционной системе Windows, а также и на Linux и Macintosh [15]. Платформа .Net Core обладает следующими особенностями:

1) поддержка нескольких языков. Основа платформы — это общеязыковая среда выполнения (CLR), из-за чего .NET поддерживает несколько языков программирования: C#, VB.NET, C++, F#. При компиляции код на любом из этих языков компилируется в сборку на общем для них языке CIL. Это, грубо говоря, ассемблер для платформы .NET. Благодаря чему мы можем комбинировать библиотеки, написанные на разных языках программирования [7];
2) кроссплатформенность. .Net Core является полностью кроссплатформенным, с некоторыми допущениями. Для полной работы всех его библиотек необходимо на компьютер установить фреймворк [6];

3) мощная библиотека классов (FCL). .Net представляет единую для всех языков библиотеку классов. И при написании любого приложения, веб-сервиса или десктопного приложения, так или иначе мы задействуем FCL;
4) разнообразные технологии. Общеязыковая среда выполнения (CLR) и библиотека классов (FCL) являются основой для всех технологий, которые можно задействовать для разработки различных приложений [14].

Например, для разработки графических приложений с богатым и красивым интерфейсом предназначена технология Windows Presentation Foundation (WPF) – это графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML [4]. В основе WPF лежит векторная система визуализации, не зависящая от разрешения устройства вывода и созданная с учётом возможностей современного графического оборудования. WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language), элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление [13].

XAML представляет собой язык декларативного описания интерфейса, основанный на XML. Также реализована модель разделения кода и дизайна, позволяющая кооперироваться программисту и дизайнеру. Кроме того, есть встроенная поддержка стилей элементов, а сами элементы легко разделить на элементы управления второго уровня, которые, в свою очередь, разделяются до уровня векторных фигур и свойств/действий. Это позволяет легко задать стиль для любого элемента, например, Button (кнопка) [10].

Графической технологией, лежащей в основе WPF, является DirectX, в отличие от Windows Forms, где используется GDI/GDI+. Производительность WPF выше, чем у GDI+ за счёт использования аппаратного ускорения графики через DirectX [3].

2.2.2. Система контроля версий Git и Source Tree

Система контроля версий – программное обеспечение для облегчения роботы с постоянно изменяющейся информацией. Эта система позволяет хранить несколько версий одного и того же документа и при необходимости можно вернутся к более ранним версиям, определять, кто и когда сделал изменения, и т.д.

Ядро Git представляет собой набор команд для командной строки с параметром. Все настройки хранятся в текстовых файлах. Такая система позволяет сделать Git легко портируемым на любую платформу и легко интегрировать в любую систему.

Репозиторий Git представляет собой каталог файловой системы, в котором находятся файлы конфигурации репозитория, файлы журналов, хранящие операции, выполняемые над репозиторием, индекс, описывающий расположение файлов, и хранилище, содержащее собственно файлы [9]. При импорте нового репозитория автоматически создается копия, соответствующая последнему состоянию репозитория на сервере.

Source Tree бесплатный git клиент для Windows. Позволяет графически увидеть представления веток в системе контроля версий. Так же упрощает работу с основными командами git: commit, push, merge, pull (рисунок 2.1).

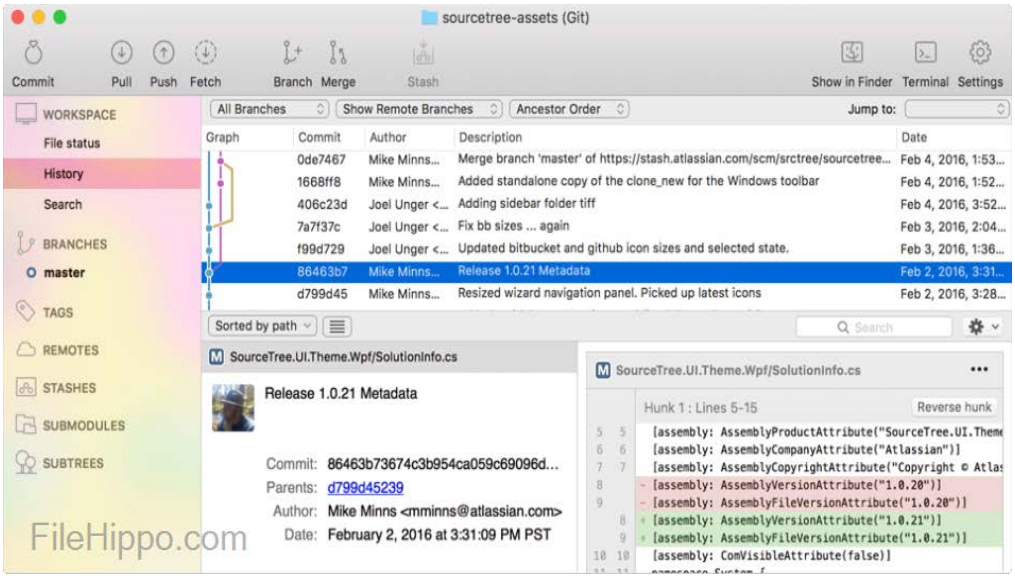

Рисунок 2.1 – Система контроля версий Source Tree

2.3. Написание алгоритма программы

Для разработки приложения фотокабины следует выделить отдельные модули, которые должны реализовывать определенную часть функций и выполнять следующие действия:

1) инициализировать камеру и подключаться к ней;
2) передавать в камеру настройки светочувствительности;
3) передавать в камеру настройки диафрагмы;
4) передавать в камеру настройки выдержки;
5) делать 3 или 4 фотографии в зависимости от выбранного шаблона;
6) вставлять полученные с камеры фотографии в шаблон;
7) отправлять итоговое изображение на печать;
8) сохранять электронные копии фотографий и шаблонов на носитель.

Приложение разрабатывается под операционную систему Windows. Для запуска необходимо иметь установленный фреймворк версии .Net Core 2 [16]. Для старта программы необходимо:

1) выбрать из списка подключенную камеру;
2) по необходимости изменить настройки светочувствительности, выдержки и диафрагмы камеры;
3) выбрать место для сохранения фотографий на жестком диске;
4) нажать на кнопку «Старт».

Основной алгоритм работы приложения представлен на рисунке 2.2.

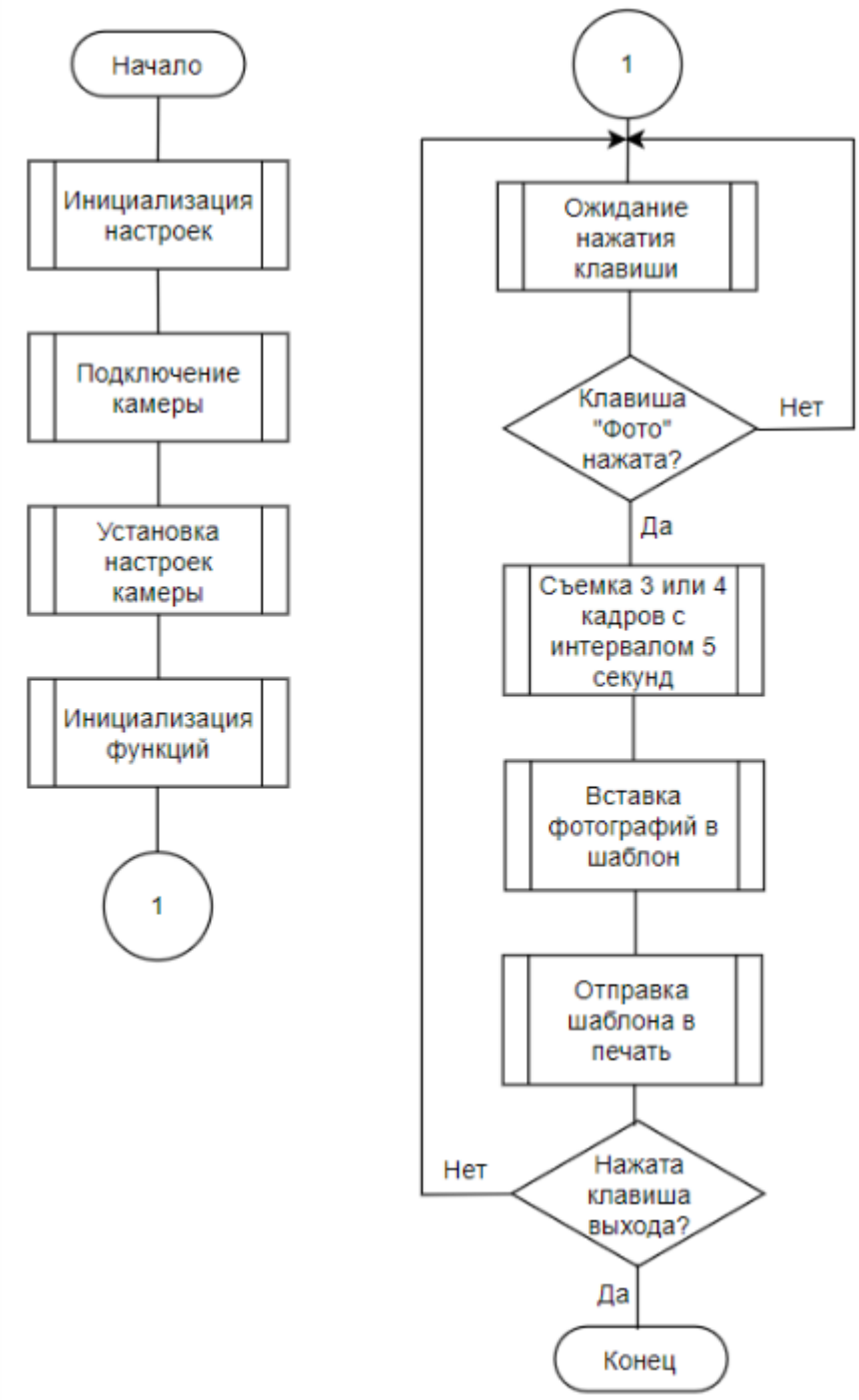

Рисунок 2.2 – Основной алгоритм программы

2.3.1. Алгоритм запуска приложения

После запуска приложения выполняется предварительная настройка и инициализация всех компонентов программы.

Далее для начала работы программы нужно выбрать в списке подключенных устройств цифровую камеру, после чего нажать кнопку «Подключить».

Затем, с помощью выпадающих списков производится настройка светочувствительности, выдержки и диафрагмы фотокамеры. После этого с помощью элемента «RadioButton» задается как будут сохраняться фотографии (в камеру, в компьютер или в оба устройства сразу), при необходимости указывается путь к паке на компьютере. По окончанию предварительной настройки нажимается кнопка «Старт». Алгоритм предварительной настройки программного обеспечения показан на рисунке 2.3.

Рисунок 2.3 – Алгоритм предварительной настройки ПО

2.3.2. Алгоритм обработки команд нажатий физических клавиш

После нажатия на клавишу «Старт» программа переходит в режим работы, транслируя видеопоток с камеры на экран фотокабины и ожидая нажатия физической кнопки на лицевой стороне корпуса. Если происходит нажатие кнопки, программа делает три или четыре фотографии с промежутком в пять секунд между каждой фотографией. После чего формируется итоговый шаблон с фотографиями и отправляется на печать в принтер. Если нажимается кнопка «Дубликат», программа отправляет на печать последний сформированный шаблон. По окончанию цикла, программа вновь переходит в режим ожидания нажатия клавиши. Данный алгоритм представлен на рисунке 2.4.

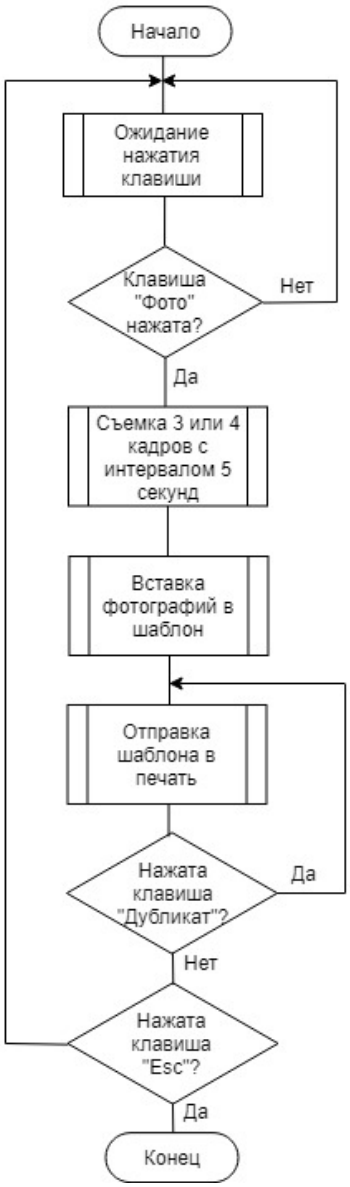

Рисунок 2.4 – Алгоритм обработки команд при нажатии на клавиши

2.4. Выводы по второй главе

В данном разделе было обосновано решение использовать C# в качестве основного языка для разработки приложения, а также фрэймворка .Net Core 2.0 для проектирования интерфейса. В качестве системы контроля версий был выбран Git под управлением программы Source Tree.

В результате были сформулированы требования к приложению и разработаны алгоритмы работы основных функций программы, а именно:

1) основной алгоритм программы;
2) алгоритм предварительной настройки ПО;
3) алгоритм обработки команд при нажатии на клавиши.

3. РАЗРАБОТКА ПРИЛОЖЕНИЯ

Все компоненты приложения реализованы на высокоуровневом языке программирования C# с поддержкой .NET Framework версии 4.5.2. Используются следующие дополнительные библиотеки классов:

1) System.Net.Http;
2) xNet;
3) Newtonsoft.Json.

Реализованы следующие алгоритмы:

1) инициализация камеры и подключение к ней;
2) передача в камеру настроек светочувствительности, диафрагмы и времени выдержки;
3) алгоритм фотографирования 3 или 4 фотографий с интервалом в 5 секунд после каждого снимка;
4) создание шаблона из полученных фотографий;
5) отправка полученного изображения на печать в принтер;
6) сохранение электронных копий фотографий и шаблонов.

3.1. Разработка графического интерфейса программы

К графическому интерфейсу предъявляется два основных требования:

1) интерфейс должен быть простой и лаконичный;
2) интерфейс должен быть интуитивно понятный.

Для разработки графического интерфейса был выбран фреймворк .NET Framework версии 4.5.2, а именно технология Windows Presentation Foundation (WPF), основанная на языке разметки XAML. Так как графический интерфейс должен удовлетворять требованию простоты и лаконичности, было принято решение использовать минимум элементов интерфейса:

1) ListBox (для вывода списка подключенных камер);
2) Button (кнопки для распределения всех основных функций);
3) GroupBox (для группировки отдельных элементов по категориям);
4) ComboBox (выпадающий список для выбора настроек камеры);

5) TextBox (для обозначения текущей директории сохранения фалов);
6) RadioButton (для выбора варианта сохранения фотографий);
7) Canvas (для вывода изображения с камеры в режиме реального времени).

Расположение элементов выбрано таким образом, чтобы они занимали минимум места, при этом логика их расположения была последовательной. То есть для того чтобы произвести первоначальную настройку программы, необходимо выполнить все необходимые действия во всех элементах интерфейса слева на право. Итоговый вид программы представлен на рисунке 3.1.

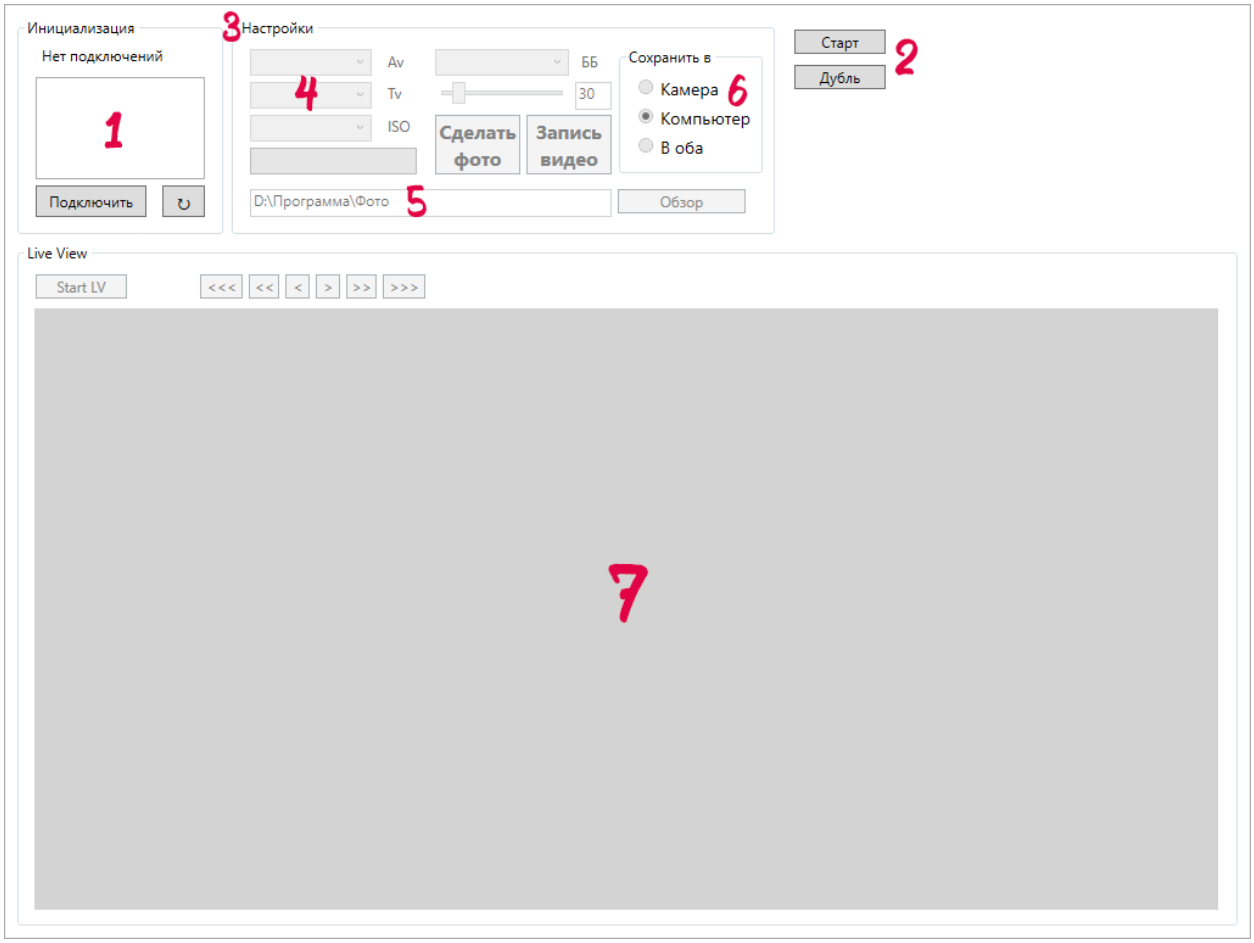

Рисунок 3.1 – Главный интерфейс программы

После того как произведена первоначальная настройка и нажата кнопка «Старт», программа переходит в режим работы и все лишние элементы интерфейса скрываются. Если во время работы появляется необходимость из-

менить некоторые настройки, скрытые элементы программы можно отобразить нажатием клавиши «Esc» на клавиатуре. Интерфейс программы в режиме работы представлен на рисунке 3.2.

Рисунок 3.2 – Интерфейс программы в режиме работы

3.2. Инициализация камеры и подключение к ней

Для того чтобы программа работала, первоначально необходимо создать функцию подключения программного обеспечения к камере. Функция реализует поиск камеры среди подключенных устройств к компьютеру и если такая находится, программа выводит данную камеру в список подключенных. После чего остается выбрать её в списке устройств и произвести подключение нажатием на кнопку «Подключить». Если же камера не обнаружена – необходимо произвести повторный поиск путем нажатия на кнопку «Обновить» в интерфейсе программы. Функция инициализации камеры и подключения к ней представлена на рисунке 3.3.

```
private void OpenSession()
{
    if (CameraListBox.SelectedIndex >= 0)
    {
        CameraHandler.OpenSession(CamList[CameraListBox.SelectedIndex]);
        SessionButton.Content = "Отключить";
        string cameraname = CameraHandler.MainCamera.Info.szDeviceDescription;
        SessionLabel.Content = cameraname;
        AvList = CameraHandler.GetSettingsList((uint)EDSDK.PropID_Av);
        TvList = CameraHandler.GetSettingsList((uint)EDSDK.PropID_Tv);
        ISOList = CameraHandler.GetSettingsList((uint)EDSDK.PropID_ISOSpeed);
        foreach (int Av in AvList) AvCoBox.Items.Add(CameraValues.AV((uint)Av));
        foreach (int Tv in TvList) TvCoBox.Items.Add(CameraValues.TV((uint)Tv));
        foreach (int ISO in ISOList) ISO-CoBox.Items.Add(CameraValues.ISO((uint)ISO));
        AvCoBox.SelectedIndex = AvCoBox.Items.IndexOf(CameraValues.AV((uint)Cam-
eraHandler.GetSetting((uint)EDSDK.PropID_Av)));
        TvCoBox.SelectedIndex = TvCoBox.Items.IndexOf(CameraValues.TV((uint)Cam-
eraHandler.GetSetting((uint)EDSDK.PropID_Tv)));
        ISOCoBox.SelectedIndex = ISO-CoBox.Items.IndexOf(CameraValues.ISO((uint)Cam-
eraHandler.GetSetting((uint)EDSDK.PropID_ISOSpeed)));
        int wbidx = (int)CameraHandler.GetSetting((uint)EDSDK.PropID_WhiteBalance);
        switch (wbidx)
        {
          case EDSDK.WhiteBalance_Auto: WBCoBox.SelectedIndex = 0; break;
          case EDSDK.WhiteBalance_Daylight: WBCoBox.SelectedIndex = 1; break;
          case EDSDK.WhiteBalance_Cloudy: WBCoBox.SelectedIndex = 2; break;
          case EDSDK.WhiteBalance_Tangsten: WBCoBox.SelectedIndex = 3; break;
          case EDSDK.WhiteBalance_Fluorescent: WBCoBox.SelectedIndex = 4; break;
          case EDSDK.WhiteBalance_Strobe: WBCoBox.SelectedIndex = 5; break;
          case EDSDK.WhiteBalance_WhitePaper: WBCoBox.SelectedIndex = 6; break;
          case EDSDK.WhiteBalance_Shade: WBCoBox.SelectedIndex = 7; break;
          default: WBCoBox.SelectedIndex = -1; break;
        }
        SettingsGroupBox.IsEnabled = true;
        LiveViewGroupBox.IsEnabled = true;
    }
}
```

Рисунок 3.3 – Инициализация камеры и подключение к ней

3.3. Загрузка настроек в камеру

После подключения камеры к программе начинается передача настроек светочувствительности, диафрагмы и выдержки, которые были установлены в качестве настроек по умолчанию ранее. После чего их можно изменить. Происходит это путем выбора из выпадающих списков различных вариантов настройки того или иного значения. Помимо функций установки светочувствительности, диафрагмы и выдержки была реализована возможность изменить настройки баланса белого. Это необходимо в тех случаях, когда источник света имеет зеленый или розовый окрас. В качестве примера можно рассмотреть реализацию функции передачи в камеру настроек светочувствительности, диафрагмы и выдержки, которая представлена на рисунке 3.4.

```
private void AvCoBox_SelectionChanged(object sender, SelectionChangedEventArgs e)
    {
      try
      {
        if (AvCoBox.SelectedIndex < 0) return;
        CameraHandler.SetSetting(EDSDK.PropID_Av, Camera-Values.AV((string)AvCoBox.Se-
lectedItem));
      }
      catch (Exception ex) { ReportError(ex.Message, false); }
    }

    private void TvCoBox_SelectionChanged(object sender, SelectionChangedEventArgs e)
    {
      try
      {
        if (TvCoBox.SelectedIndex < 0) return;

        CameraHandler.SetSetting(EDSDK.PropID_Tv, Camera-Values.TV((string)TvCoBox.Se-
lectedItem));
        if ((string)TvCoBox.SelectedItem == "Bulb")
        {
          BulbBox.IsEnabled = true;
          BulbSlider.IsEnabled = true;
        }
        else
        {
          BulbBox.IsEnabled = false;
          BulbSlider.IsEnabled = false;
        }
      }
      catch (Exception ex) { ReportError(ex.Message, false); }
    }

    private void ISOCoBox_SelectionChanged(object sender, SelectionChangedEventArgs e)
    {
      try
      {
        if (ISOCoBox.SelectedIndex < 0) return;
        CameraHandler.SetSetting(EDSDK.PropID_ISOSpeed, Camera-Values.ISO((string)ISO-
CoBox.SelectedItem));
      }
      catch (Exception ex) { ReportError(ex.Message, false); }
    }
```

Рисунок 3.4 – Передача в камеру настроек

3.4. Обработка команд при нажатии на клавишу

После нажатия на кнопку «Старт» в графическом интерфейсе, программа переходит в режим работы и ожидает нажатия на одну из двух физических клавиш, находящихся на лицевой стороне фотокабины. Реализованный алгоритм отслеживает нажатие на клавишу и выполняет цикл фотографирования трех или четырех фотографий с интервалом в пять секунд между каждым снимком. Значение интервала между фото задается в коде функции. Алгоритм обработки нажатия на клавишу представлен на рисунке 3.5.

```
public void _fotobutton()
{
  if (pressed == 0)
  {
    pressed = 1;
    int b, r, r2;
    r = 5; //Счетчик
    eh = (object mySender, EventArgs args) => //прописываем событие
      {
        if (r >= 1) //Если счетчик меньше 5, то выводим картинку
        {
          _imgload("" + r);
        }
        else
        {
          timer.Stop();
          CameraHandler.TakePhoto();
          r2 = 3;
          seriaphoto += 1;
          timer2.Start();
        }
        r -= 1; // Прибавляем к счетчику 1
      };
    timer.Tick += eh;
    timer.Start();
    #endregion
    #region Показ комментариев
    timer2.Interval = new TimeSpan(0, 0, 0, 1, 0); //Интервал 1
секунда
    EventHandler eh2 = null; //Создаем событие
    r2 = 3; b = 1;//Время таймера
    eh2 = (object mySender, EventArgs args) => //прописываем событие
      {
        if (r2 >= 1)
        {
          if (b != 0)
          {
            int c = rnd.Next(1, 3);
            _imgload("comment" + c);
            b = 0;
          }
        }
        else
        {
          if (seriaphoto == 3)
          {
            timer2.Stop();
            //MessageBox.Show("Серия фотографий сделана!");
            seriaphoto = 0;
            _collage();
            _imgload("finish");
            _printcollage();
            _movefile();
          }
          else { timer2.Stop(); r = 5; b = 1; timer.Start(); }
        }
        r2 -= 1; // Прибавляем к счетчику 1
      };
    timer2.Tick += eh2;
  }
}
```

Рисунок 3.5 – Обработка команд при нажатии на клавишу

После съемки трех или четырех снимков в работу вступает реализованный алгоритм создания шаблона, который представлен на рисунке 3.6.

```
private void _collage()
  {
    string[] files = Directory.GetFiles(@SavePathTextBox.Text);
    BitmapImage photo1 = new BitmapImage();
    photo1.BeginInit();
    photo1.CacheOption = BitmapCacheOption.OnLoad;
    photo1.UriSource = new Uri(@files[0]);
    photo1.EndInit();
    okno1.Source = photo1;
    BitmapImage photo2 = new BitmapImage();
    photo2.BeginInit();
    photo2.CacheOption = BitmapCacheOption.OnLoad;
    photo2.UriSource = new Uri(@files[1]);
    photo2.EndInit();
    okno2.Source = photo2;
    BitmapImage photo3 = new BitmapImage();
    photo3.BeginInit();
    photo3.CacheOption = BitmapCacheOption.OnLoad;
    photo3.UriSource = new Uri(@files[2]);
    photo3.EndInit();
    okno3.Source = photo3;
    BitmapImage photo4 = new BitmapImage();
    photo4.BeginInit();
    photo4.CacheOption = BitmapCacheOption.OnLoad;
    photo4.UriSource = new Uri(@files[0]);
    photo4.EndInit();
    okno4.Source = photo4;
    BitmapImage photo5 = new BitmapImage();
    photo5.BeginInit();
    photo5.CacheOption = BitmapCacheOption.OnLoad;
    photo5.UriSource = new Uri(@files[1]);
    photo5.EndInit();
    okno5.Source = photo5;
    BitmapImage photo6 = new BitmapImage();
    photo6.BeginInit();
    photo6.CacheOption = BitmapCacheOption.OnLoad;
    photo6.UriSource = new Uri(@files[2]);
    photo6.EndInit();
    okno6.Source = photo6;
    itmapImage shablon = new BitmapImage();
    shablon.BeginInit();
    shablon.CacheOption = BitmapCacheOption.OnLoad;
    shablon.UriSource = new Uri(Directory.GetCurrentDirectory() +
@"\Шаблоны\1.png");
    shablon.EndInit();
    fon.Source = shablon;
    _SaveCollage(new Uri(@SavePathTextBox.Text + @"\Склейки\" + photocount +
".jpg"), imgprint);
  }
```

Рисунок 3.6 – Вставка полученных фотографий в шаблон

После получения итогового изображения, оно передается на принтер, где в свою очередь распечатывается и вручается клиенту. Реализованный алгоритм печати шаблона представлен на рисунке 3.7.

```
private void _printcollage()
{
  var bi = new BitmapImage();
  bi.BeginInit();
  bi.CacheOption = BitmapCacheOption.OnLoad;
  bi.UriSource = new Uri(@SavePathTextBox.Text + @"\Склейки\" + photocount +
".jpg");
  bi.EndInit();
  var vis = new DrawingVisual();
  var dc = vis.RenderOpen();
  dc.DrawImage(bi, new Rect { Width = bi.Width / 3.080, Height = bi.Height /
3.118, X = 8, Y = 8 });
  dc.Close();
  var pdialog = new PrintDialog();
  pdialog.PrintVisual(vis, photocount+" - Фотобудка ВВОХ");
  photocount += 1;
  fon.Source = null;
  fon2.Source = null;
  okno1.Source = null;
  okno2.Source = null;
  okno3.Source = null;
  okno4.Source = null;
  okno5.Source = null;
  okno6.Source = null;
  okno7.Source = null;
  okno8.Source = null;
  okno9.Source = null;
  okno10.Source = null;
}
```

Рисунок 3.7 – Формирование изображения и отправка его на печать

3.5. Тестирование программы

Тестирование проводилось в условиях реальной работы на одном из мероприятий города, программа была запущена на локальной машине с установленным на ней фрэймворком .NET Framework версии 4.5.2. Компьютер имел следующие характеристики:

1) процессор Intel(R) Core(TM) i7 CPU 870 2.93Ghz;
2) оперативная память 8,0 ГБ;
3) дисковое устройство SSD Kingston 120 ГБ.

Сценарий тестирования:

1) запуск программы;
2) задание первоначальных настроек камеры;
3) запуск пользовательского интерфейса;
4) нажатие на клавишу фотографирования на корпусе фотокабины;
5) цикл из 3 фотографий с паузой в 5 секунд после каждого фото;

6) получение готовой фотографии;
7) печать дубликата путем нажатия на кнопку «Дубликат»;
8) получение второй готовой фотографии;
9) проверка программы на стабильность, путем непрерывной работы в течении 2 часов;
10) выявление и фиксирование недостатков программы.

В результате тестирования было выяснено, что программа ведет себя стабильно даже при условиях режима работы без остановки. Недостатков в работе программного обеспечения выявлено не было.

3.6. Выводы по третьей главе

В третьей главе была рассмотрена реализация графического интерфейса, а также основных функций программного обеспечения, спроектированных во второй главе.

В результате разработки было произведено тестирование программного обеспечения на стабильность работы в условиях беспрерывной работы в течении двух часов.

На основе полученных данных был сделан вывод, что результаты тестирования полностью соответствуют необходимым требованиям.

ЗАКЛЮЧЕНИЕ

В данной выпускной квалификационной работе был рассмотрен процесс создания приложения для автоматизации управления процессами фотокабины. Были проанализированы готовые решения поставленной задачи, в следствии чего было принято решение разрабатывать программное обеспечение самостоятельно.

В ходе работы был выбран язык программирования C#, а также средства проектирования и отладки. После чего был разработан графический интерфейс, а также основные алгоритмы работы программного обеспечения.

По окончанию разработки, приложение было протестировано на предмет стабильности в условиях непрерывной работы в течении двух часов на одном из мероприятий города Челябинск.

В результате тестирования был сделан вывод о том, что программа отлично справляется с поставленной задачей в условиях непрерывной работы, что в свою очередь свидетельствует о том, что программное обеспечение написано на высоком уровне, а все возможные ошибки были учтены на этапе проектирования.

СПИСОК ЛИТЕРАТУРЫ

1. Голдовский Е.М. Основы кинотехники. / Е.М. Голдовский, Л.О. Эйсымонт. – М.: Искусство, 1965. – 636 с.

2. Иофис Е.А. Фотокинотехника / И. Ю. Шебалин. – М.: Советская энциклопедия, 1981. –355 с.

3. Ликнесс Д. Приложения для Windows 8 на C#. / Д. Ликнесс. –М.: Питер, 2013. – 368 с.

4. Мак-Дональд М. WPF: Windows Presentation Foundation в .NET 4.5 с примерами на C# 5.0 для профессионалов. 4-е изд. / М.Мак-Дональд – М.: Вильямс, 2013. –1024 с.

5. Панфилов Н.Д. Краткий справочник фотолюбителя / Н. Д. Панфилов, А.А. Фомин. – М.: Искусство, 1985. – 367 с.

6. Скит Д. C# для профессионалов: тонкости программирования, 3-е издание, новый перевод. — М.: Вильямс, 2014. – 608 с.

7. Троелсен Э. Язык программирования C# 5.0 и платформа .NET 4.5. 6-е изд. /Э. Троелсен – М.: Вильямс, 2015. –1312 с.

8. Уэстон К. Экспозиция в цифровой фотосъёмке. / Т.И. Хлебнова. – М.: АРТ-родник, 2008. – 192 с.

9. Чакон С. Git для профессионального программиста. / С. Чакон, Б. Штрауб. – М.: Питер, 2017. – 496 с.

10. draw.io, доступ: https://www.draw.io/ (дата обращения – 3 мая 2019).

11. lux-foto.com, доступ: http://lux-foto.com/pages/software/photobox/ (дата обращения – 28 апреля 2019).

12. lux-foto.zendesk.com, доступ: https://lux-foto.zendesk.com/hc/ru/articles/206175921-описание-интерфейса-1 (дата обращения – 29 апреля 2019).

13. metanit.com, доступ: https://metanit.com/sharp/wpf/ (дата обращения – 12 апреля 2019).

14. microsoft.com, доступ: https://dotnet.microsoft.com/apps/aspnet (дата обращения – 7 апреля 2019).

15. microsoft.com, доступ: https://docs.microsoft.com/ru-ru/dotnet/csharp/ (дата обращения – 15 марта 2019).

16. microsoft.com, доступ: https://dotnet.microsoft.com/learn/dotnet/what-is-dotnet/ (дата обращения – 3 февраля).

17. openbusiness.ru, доступ: https://www.openbusiness.ru/html/dop5/mom.htm (дата обращения – 15 января 2019).

18. photoboothsolutions.com, доступ: http://www.photoboothsolutions.com/socialbooth/ (дата обращения – 22 мая 2019).

19. photofabrika.ru, доступ: https://photofabrika.ru/soft/photobooth/ (дата обращения – 22 мая 2019).

20. photostarpro.ru, доступ: https://photostarpro.ru/2019/04/15/ne-znaete-chto-takoe-fotobudka/ (дата обращения – 3 июня 2019).