|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

|  |  |
| --- | --- |
| **ФАКУЛЬТЕТ** | ***ИУК «Информатика и управление»*** |
| **КАФЕДРА** | ***ИУК4 «Программное обеспечение ЭВМ,*** |
|  | ***информационные технологии»*** |

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе на тему:**

|  |
| --- |
| ***Разработка банковского программного обеспечения*** |
|  |

|  |  |
| --- | --- |
| по дисциплине | *Базы данных* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Студент гр. | *ИУК4-62Б* | |  |  | ( | *Калашников А.С.* | ) |
|  | |  |  | (подпись) |  | (Ф.И.О.) |  |
| Руководитель | |  |  |  | ( | *Глебов С.А.* | ) |
|  | |  |  | (подпись) |  | (Ф.И.О.) |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Оценка руководителя |  | баллов |  |  |
|  | 30-50 |  | (дата) |  |
| Оценка защиты |  | баллов |  |  |
|  | 30-50 |  | (дата) |  |
| Оценка проекта |  | баллов |  | |
|  |  |  | (оценка по пятибалльной шкале) | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Комиссия: |  |  | ( | *Глебов С.А.* | ) |
|  |  | (подпись) |  | (Ф.И.О.) |  |
|  |  |  | ( | *Гришунов С.С.* | ) |
|  |  | (подпись) |  | (Ф.И.О.) |  |
|  |  |  | ( | *Гагарин Ю.Е.* | ) |
|  |  | (подпись) |  | (Ф.И.О.) |  |

Калуга, 2023

Калужский филиал   
федерального государственного бюджетного образовательного учреждения высшего образования

***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»   
(КФ МГТУ им. Н.Э. Баумана)***

УТВЕРЖДАЮ

Заведующий кафедрой *ИУК4*

( *Гагарин Ю.Е.)*

« *09* » *февраля* 20*23* г.

**ЗАДАНИЕ**

**на выполнение курсовой работе**

|  |  |
| --- | --- |
| по дисциплине | ***Базы данных*** |

|  |  |  |
| --- | --- | --- |
| Студент | *Калашников А.С. ИУК4-62Б* | |
|  | | (фамилия, инициалы, индекс группы) |
| Руководитель | | *Глебов С.А.* |
|  | | (фамилия, инициалы) |

График выполнения работы: 25% к *4* нед., 50% к *7* нед., 75% к *10* нед., 100% к *14*  нед.

|  |
| --- |
| ***1. Тема курсовой работы*** |
| Разработка банковского программного обеспечения |
|  |
|  |
| ***2. Техническое задание*** |
| Разработать набор приложений обеспечивающий комфортное взаимодействие |
| пользователей и сотрудников банка |
|  |

|  |
| --- |
| ***3. Оформление курсовой работы*** |
| 3.1. Расчетно-пояснительная записка на  *XX*  листах формата А4. |
| 3.2. Перечень графического материала КП (плакаты, схемы, чертежи и т.п.) |
| *1. Диаграмма классов* |
| *2. Демонстрационный чертеж* |
|  |
|  |

Дата выдачи задания « *09* » *февраля* *2023* г.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Руководитель курсовой работы | | |  | | / | *Глебов С.А.* |
|  | | (подпись) | | |  | (Ф.И.О.) |
| Задание получил |  | | / | *Калашников А.С.* | / | « *09* » *февраля* 20*23* г. |
|  | (подпись) | |  | (Ф.И.О.) |  |  |

СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 5](#_Toc89360893)

[ВВЕДЕНИЕ 6](#_Toc89360894)

[1. МЕТОДЫ И ИНСТРУМЕНТЫ ПРОГРАММНОЙ ИНЖЕНЕРИИ 8](#_Toc89360895)

[1.1. Техническое задание 8](#_Toc89360896)

[1.2. Анализ существующих аналогов 14](#_Toc89360897)

[1.3. Перечень задач, подлежащих решению в процессе разработки. 15](#_Toc89360898)

[1.4. Обоснование выбора ОС для разработки 15](#_Toc89360899)

[1.5. Обоснование выбора СУБД 17](#_Toc89360900)

[1.6. Обоснование выбора языков программирования 20](#_Toc89360901)

[1.7. Обоснование выбора сред разработки 22](#_Toc89360902)

[1.8. Описание реализуемой архитектуры клиент-сервер 24](#_Toc89360903)

[1.9. Выбор средства для взаимодействия с базой данных 24](#_Toc89360904)

[1.10. Информация о QR-коде 25](#_Toc89360905)

[Выводы 26](#_Toc89360906)

[2. ПРОЕКТИРОВАНИЕ КОМПОНЕНТОВ ПРОГРАММНОГО ПРОДУКТА 28](#_Toc89360907)

[2.1. Разработка структуры системы 28](#_Toc89360908)

[2.2. Разработка базы данных 28](#_Toc89360909)

[2.3. Построение диаграммы вариантов использования Windows приложения 31](#_Toc89360910)

[2.4. Структура Android приложения 32](#_Toc89360911)

[2.5. Организация сетевого взаимодействия. 37](#_Toc89360912)

[Выводы 40](#_Toc89360913)

[3. ТЕСТИРОВАНИЕ И ИНТЕГРАЦИЯ КОМПОНЕНТОВ ПРОГРАММНОГО   
ПРОДУКТА 41](#_Toc89360914)

[3.1. Тестирование системы 41](#_Toc89360915)

[3.2. Руководство администратора 42](#_Toc89360916)

[3.3. Руководство пользователя 43](#_Toc89360917)

[Выводы 54](#_Toc89360918)

[ЗАКЛЮЧЕНИЕ 55](#_Toc89360919)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 56](#_Toc89360920)

[Основная литература 56](#_Toc89360921)

[Дополнительная литература 56](#_Toc89360922)

ВВЕДЕНИЕ

Курсовой работы по созданию банковского программного обеспечения является актуальной темой в современном мире, так как банки активно развиваются и используют новые технологии для улучшения своей деятельности. Кроме того, традиционные банковские услуги становятся все более цифровыми и современными, поэтому банковское ПО становится необходимым инструмент.

Объектом курсовой работы по созданию банковского программного обеспечения является система, состоящая из отдельных программ, которые обеспечивают эффективную работу банка в целом.

Предметом исследования курсовой работы по созданию банковского программного обеспечения являются различные аспекты разработки ПО для банковской сферы, включая определение требований к создаваемому ПО, проектирование архитектуры системы, выбор используемых технологий и основных модулей, создание базы данных, написание программного кода, тестирование и внедрение программного продукта в банковскую инфраструктуру.

Целью курсовой работы по созданию банковского программного обеспечения является разработка функционально полного программного продукта, который будет обеспечивать эффективное управление финансовыми процессами в банке и удобство использования услуг банка для клиентов.

Для достижения поставленной цели решаются следующие задачи:

1. Выполнить анализ предметной области
2. Провести сравнительный анализ существующих аналогов
3. Определить оптимальную структуру системы
4. Осуществить выбор средств реализации программного продукта, соответствующего выбранной структуре.
5. Реализовать схему базы данных
6. Реализовать базу данных и программные компоненты системы
7. Осуществить тестирование компонентов
8. Разработать сопроводительную документацию

1. МЕТОДЫ И ИНСТРУМЕНТЫ ПРОГРАММНОЙ ИНЖЕНЕРИИ

1.1. Техническое задание

Наименование системы

Настоящее Техническое задание определяет требования и порядок создания банковского программного обеспечения.

Основания для разработки

Основанием для разработки банковского программного обеспечения является необходимость обновления и улучшения банковских услуг, а также растущие потребности клиентов в удобном и быстром взаимодействии с банком. Современная жизнь требует быстрого и качественного обслуживания, а развитие технологий позволяет создать продукты и сервисы, которые максимально соответствуют потребностям клиентов.

Кроме того, банки действуют в условиях жесткой конкуренции, и чтобы сохранять свои позиции и привлекать новых клиентов, им необходимо обеспечить удобные и современные сервисы, использующие все новые технологии.

Разработка банковского программного обеспечения позволяет автоматизировать процессы, связанные с обслуживанием клиентов, услугами платежей, кредитования и другими банковскими операциями, а также обеспечить более точный и удобный контроль за всеми финансовыми потоками. Все это помогает банкам повысить свою эффективность и конкурентоспособность на рынке.

Исполнитель

Исполнителем проекта является студент Калужского филиала МГТУ им. Н. Э. Баумана, факультета ИУК, группы ИУК4-62Б, Калашников Артем Сергеевич.

Краткая характеристика области применения

Разрабатываемая система предназначена для автоматизации процессов управления регистрационными записями клиентов, обработки платежей, управления оборотным капиталом и денежными потоками, а также для контроля за финансовыми операциями и проведения аналитики деятельности банка.

Целевая аудитория

Инженерно-технический персонал и клиенты.

Назначение приложений

Основное назначение банковской системы заключается в обеспечении как следующих основных задач:

1. предоставление услуг клиентам: банковская система позволяет клиентам получать доступ к банковским услугам, таким как открытие счетов, выдача кредитов, проведение платежных операций и т.д.;
2. управление финансовыми ресурсами банка: банковская система позволяет банку эффективно управлять своими ресурсами, включая мониторинг и управление денежными потоками, управление кредитным портфелем, управление рисками и т.д.;
3. обработка финансовых операций: банковская система обеспечивает обработку и учет финансов

Цели разработки

Целью создания банковской системы являются оптимизация работы банка и улучшение качества обслуживания клиентов.

Плановые сроки начала и окончания работы по созданию системы

Планируемые сроки начала и окончания работы над проектом: 09.02.2023 – 28.04.2023.

Требования к приложениям

Разрабатываемая система должна являться набором приложений для взаимодействия клиента и банка и должна иметь следующий функционал:

1. Регистрация и авторизация пользователя;
2. Автоматическое добавление каждому клиенту базового счета
3. Просмотр информации о пользователе;
4. Добавление нового счета;
5. Перевод денежных средств
6. Просмотр недавних взаимодействий по счету
7. История по счету
8. Вкладывания денежных средств по определенным условиям
9. Добавление администратором новые вклады с различными условиями

Требования к временным характеристикам

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать времени восстановления операционной системы и восстановления работы сети.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Этапы разработки

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

На стадии проектирования программы должен быть выполнен этап выборки программного обеспечения, библиотек для создания, этап проектирования системы в целом, разработка технической документации.

На стадии реализации производится разработка и тестирование спроектированного приложения.

1.2. Постановка задачи

Задачей данной курсовой работы является разработка банковской системы, которая обеспечивает эффективное управление финансовыми процессами в банке и удобство использования услуг банка для клиентов.

1.3. Анализ существующих аналогов

В настоящее время существуют аналоги предоставляющие подобный функционал. Однако данные системы являются дорогостоящими и в дополнение к актуальному перечню задач предоставляют множество других, которые зачастую являются избыточными и только нагружают, и усложняют систему, а также требуют больших аппаратных ресурсов.

1.4. Перечень задач, подлежащих решению в процессе разработки.

Для выполнения поставленной задачи необходимо определить оптимальную архитектуру для построения системы и выбрать подходящие средства разработки для данной информационной системы.

Структура системы должна включать в себя сервер и набор клиентов. Серверная часть состоит из БД с информацией об оборудовании и приложения для работы с БД. Клиентская часть будет реализована в виде мобильного приложения, которое позволить получать данные от сервера, путем передачи ему информации, зашифрованной в QR коде оборудования и генерировать такие коды при необходимости. Средства реализации включают в себя: СУБД для реализации базы данных, языки программирования для реализации серверного и клиентского приложений, ОС для разработки, среды разработки, инструмент взаимодействия между клиентами и сервером базы данных.

1.5. Обоснование выбора СУБД

Общие сведения

Разрабатываемая система предполагает хранение и обработку данных, следовательно, необходимо средство для эффективного и удобного выполнения этих задач. Наилучшим образом для этого подходят базы данных.

Различают три основных типа баз данных: иерархическую, сетевую и реляционную. В настоящее время иерархические и сетевые модели данных уступают реляционной. Такое положение обосновано рядом имеющихся у реляционной модели преимуществ, среди которых: простота и наглядность в сравнении с другими двумя моделями, удобство физической реализации на ЭВМ, применение строгих правил при проектировании базирующихся на математическом аппарате, независимость данных.

Для взаимодействия с базами данных используются системы управления базами данных (СУБД). Для разработки базы данных было необходимо выбрать наиболее подходящую СУБД для поставленной задачи. В силу распространенности, для рассмотрения были выбраны: Firebird, MySQL, SQLite, PostgreSQL.

СУБД PostgreSQL

Свободно распространяемая СУБД, относится к объектно-реляционному типу. PostgreSQL отличается тем, что предоставляет объектно-ориентированный функционал, в том числе обладает поддержкой концепта ACID (атомарность, непротиворечивость, изолированность, сохранность данных).

Преимущества:

* Полная SQL-совместимость.
* Расширяемость: PostgreSQL —можно программно расширить за счёт хранимых процедур.
* Объектно-ориентированность: PostgreSQL — не только реляционная, но и объектно-ориентированная СУБД.

Недостатки:

* Производительность: в простых операциях чтения PostgreSQL может уступать своим соперникам, таким как MySQL.
* Популярность: из-за своей сложности инструмент не очень популярен.
* Подходит больше для работы со сложными структурами данных и не менее сложными процедурами.

СУБД SQLite

SQLite - встраиваемая компактная реляционная база данных.

Особенность SQLite в том, что она не использует парадигму [клиент-сервер](http://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80), поэтому её и называют «встраиваемой». Движок SQLite - не отдельно работающий процесс, а библиотека, с которой программа компонуется и движок становится составной частью программы.

Преимущества:

* Файловая: вся база данных хранится в одном файле, что облегчает перемещение.
* Стандартизированная: SQLite использует SQL.
* Отличная при разработке и тестировании - в процессе разработки приложений часто появляется необходимость масштабирования. SQLite предлагает всё что необходимо для этих целей, так как состоит всего из одного файла и библиотеки написанной на языке C.

Недостатки:

* Отсутствие системы пользователей. Если необходимо обеспечить доступ к данным для нескольких пользователей, и к тому же различать их по правам доступа, то стоит выбрать другую СУБД.
* Отсутствие возможности увеличения производительности.
* Невозможность записи больших объемов данных - одним из ограничений SQLite являются операции записи.

СУБД FireBird

FireBird – свободно распространяемая СУБД. Поддерживает стандарты ANSI в синтаксисе языка SQL и позволяет работать на многих операционных системах.

Преимущества:

* [Многоверсионная архитектур](http://ru.wikipedia.org/wiki/MVCC)а (параллельная обработка оперативных и аналитических запросов: читающие пользователи не блокируют пишущих);
* Соответствие требованиям [ACID](https://ru.wikipedia.org/wiki/ACID).

Недостатки:

* Отсутствие кеша итогов запросов, полнотекстовых индексов.
* Значительное падение производительности при росте внутренней фрагментации базы.

СУБД MySQL

Данная СУБД отличается большой скоростью, устойчивостью и легкостью использования. Предлагает широкий набор инструментов для разработки приложений. Благодаря своей популярности, существует много различных плагинов и расширений, которые могут упростить работу с системой. В силу того, что MySQL является серверной СУБД, приложения для доступа к данным, в отличии от SQLite работают со службами MySQL. Является идеальным решением для малых и средних приложений. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

Преимущества:

* Простота: MySQL легко устанавливается. Существует много сторонних инструментов, облегчающих работу с БД.
* Широкий функционал: MySQL поддерживает большую часть функционала SQL.
* Безопасность: большое количество функций, обеспечивающих безопасность, которые поддерживается по умолчанию.
* Масштабируемость: MySQL легко работает с большими объемами данных и легко масштабируется.
* Скорость: упрощение некоторых стандартов позволяет MySQL значительно увеличить производительность.
* Поддерживается неограниченное количество пользователей, одновременно работающих с базой данных, а количество строк в таблицах может достигать 50 млн.

Недостатки:

* Медленная разработка приложений в MySQL, о чем свидетельствуют негативные отзывы.
* Застой в разработке: хотя MySQL и является open-source продуктом, работа над ней сильно заторможена.

1.6. Обоснование выбора языков программирования

**Для Windows приложения**

C# - [объектно-ориентированный](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) [язык программирования](http://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) для платформы .NET. Основным постулатом С# является высказывание: "всякая сущность есть объект". Язык основан на строгой компонентной архитектуре и реализует передовые механизмы обеспечения безопасности кода.

C# относится к семье языков с C-подобным синтаксисом. Поддерживает черты объектно-ориентированного программирования. Является надежным и устойчивым языком, за счет использования «сборки мусора», безопасности типов и обработки исключений. C# разрабатывался "с нуля" и вобрал в себя много полезных свойств таких языков, как C++, Java, Visual Basic, а также Pascal, Delphy и др., избежав при этом многих отрицательных сторон своих предшественников. Мощная библиотека каркаса Framework.Net поддерживает удобство построения различных типов приложений и позволяет достаточно просто сохранять и получать информацию из баз данных.

**Для мобильного приложения**

Выбор языка программирования для разработки мобильного приложения во многом зависит от поставленных целей. Один из критериев для определения подходящего языка программирования является ОС под которую планируется разработка приложения. Для ОС Android одними из самых популярных языков являются: HTML5/JavaScript, Java, C/С++.

**HTML5/JavaScript** - являются сценарными языками программирования, адаптированными для разработки мобильных веб-приложений. Они несомненно обладают рядом преимуществ, например, кроссплатформенность, придание интерактивности веб-страницам. Поскольку разрабатываемое приложение является видом нативного приложения с тесным использованием функций ПО смартфорна (камеры), данный стандарт не подходит для поставленной задачи.

**C/C++** - нативные языки программирования, которые за счет использования NDK, также можно использовать для разработки под ОС Android.Данные языки, несомненно, не теряют своей актуальности и хорошо подходят для низкоуровневого программирования, позволяя создавать более производительные, быстрые, ресурсоемкие приложения. Код является более быстрым, за счет отсутствия необходимости использования виртуальной машины, но данное преимущество является существенным недостатком, сказывающимся в отсутствии переносимости. К тому же отсутствует объектно-ориентированный подход, что усложняет понимание и контроль кода.

**Java** - считается основным и официальным языком для программирования под ОС Android. Java является объектно-ориентированным, строго типизированным языком программирования, что дает ему преимущество перед другими средствами разработки. В связи с широким его применением, имеется множество руководств, учебников, инструкций, что может помочь при поиске необходимой информации или решения, при возникновении трудной ситуации. Java поддерживает множество API, что предоставляет широкий выбор инструментов для решений различных задач, включая актуальную для поставленной задачи использование возможностей камеры и обработки полученных изображений. К тому же Java используется в ряде мощных, современных интегрированных средств разработки (IDE), применение которых делает разработку более простой и контролируемой.

1.6. Обоснование выбора сред разработки графического интерфейса

**PyQt** - это графический пользовательский интерфейс (GUI) для Python. Он очень популярен среди разработчиков, и графический интерфейс может быть создан с помощью кодирования или QT-дизайнера. Фреймворк разработки QT-это визуальный фреймворк, который позволяет перетаскивать виджеты для создания пользовательских интерфейсов.

Это бесплатное программное обеспечение с открытым исходным кодом для привязки и реализовано для кросс-платформенной платформы разработки приложений. Он используется в Windows, Mac, Android, Linux и Raspberry PI.

PyQt-это зрелый набор привязок Python к Qt для кросс-платформенной разработки настольных приложений. Он предлагает богатый выбор встроенных виджетов и инструментов для создания пользовательских виджетов для формирования сложных графических интерфейсов, а также надежную поддержку баз данных SQL для подключения к базам данных и взаимодействия с ними.

**Tkinter** – одна из самых популярных библиотек графического интерфейса Python для разработки настольных приложений. Это комбинация стандартного графического интерфейса TK и python.

Tkinter предоставляет различные виджеты , такие как метки , кнопки, текстовые поля, флажки, которые используются в приложении с графическим пользовательским интерфейсом.

Виджеты управления кнопками используются для отображения и разработки приложений, в то время как виджет canvas используется для рисования фигур, таких как линии, полигоны, прямоугольники и т. Д. В приложении. Кроме того, Tkinter-это встроенная библиотека для Python, поэтому вам не нужно устанавливать ее, как другие графические фреймворки.

**Kivy** - это библиотека Python с открытым исходным кодом для быстрой разработки приложений, использующих инновационные пользовательские интерфейсы, такие как мультитач-приложения.

Kivy работает на Linux, Windows, OS X, Android, iOS и Raspberry Pi. Вы можете запустить один и тот же код на всех поддерживаемых платформах. Он может изначально использовать большинство входов, протоколов и устройств, включая WM\_Touch, WM\_Penn, трекпад Mac OS X и Magic Mouse, Netdev, ядро Linux HID.

Kivy на 100% бесплатен в использовании по лицензии MIT.

Инструментарий профессионально разработан, поддерживается и используется. Вы можете использовать его в коммерческом продукте. Фреймворк стабилен и имеет хорошо документированный API, а также руководство по программированию, которое поможет вам начать работу.

Графический движок Kivy построен на OpenGL ES 2, используя современный и быстрый графический конвейер.

Инструментарий поставляется с более чем 20 виджетами, все они очень расширяемы. Многие части написаны на языке Си с использованием Cython и протестированы с помощью регрессионных тестов

**wxPython** - это кроссплатформенный графический инструментарий для языка программирования Python.

Это позволяет программистам на Python создавать программы с надежным, высокофункциональным графическим пользовательским интерфейсом, просто и легко. Он реализован в виде набора модулей расширения Python, которые обертывают компоненты графического интерфейса популярной кроссплатформенной библиотеки wxWidgets, написанной на C++.

Как и Python и wxWidgets, wxPython является открытым исходным кодом.

wxPython-это кроссплатформенный инструментарий. Это означает, что одна и та же программа будет работать на нескольких платформах без изменений. В настоящее время поддерживаются следующие платформы: Microsoft Windows, Mac OS X и macOS, а также Linux.

1.7. Обоснование выбора сред разработки

Данный выбор оказывает значительное влияние на удобство разработки, имеющийся инструментарий, возможности отладки и развертывания приложений на различных платформах, эффективность и качество реализуемых приложений.

Самым популярным решением для разработки под ОС Windows является среда Microsoft Visual Studio. Данная среда содержит компилятор языка программирования C#, предоставляет широкие возможности по отладке программ, включает в себя редактор исходного кода с поддержкой технологии IntelliSense, а также, что самое важное поддерживает технологию Windows Forms, с помощью которой предполагается разработка графического интерфейса приложения.

В вопросе разработки мобильного приложения необходимо учесть, что разработка предполагается на языке Java для ОС Android. Самыми популярными являются следующие IDE: NetBeans, Eclipse, Intellij IDEA и, основанная на предыдущей, Android Studio.

**Eclipse** долгое время была основной средой для разработки приложений под Android. В связи с этим имеется множество ресурсов по вопросам Eclipse, видео, уроков, примеров, статей. Все это может оказать значительную помощь при разработке. Обладает стабильным, широким функционалом и развернутой системой плагинов. Однако имеется высокая вероятность столкнуться с ошибками несовместимости при установки этих плагинов. Не смотря на преимущества, по удобству данная IDE уступает не менее популярной Intellij IDEA.

**Intellij IDEA** – среда разработки, альтернативная Eclipse. Развитием занимается компания JetBrains. Компания постоянно занимается её расширением, обновлением с целью соответствовать самым последним потребностям рынка. Обладает большими возможностями и функционалом, удобством разработки, инструментами отладки и рефакторинга кода.

**NetBeans** – среда разработки, которая во многом не уступает другим IDE. Содержит много средств для редактирования, рефакторинга кода, включая «умное» автодополнение кода, присущее так же рассмотренным Intellij IDEA и Eclipse. Причем справляется с этой задачей несколько быстрее чем Eclipse, но медленнее чем, Intellij IDEA. Несмотря на всю свою мощь и перспективность NetBeans обладает существенным недостатком. Данная IDE является менее распространенной средой разработки, в связи с этим имеется меньше доступных источников информации.

**Android Studio** – развитие Intellij IDEA, разрабатываемая Google. Компания Google включила в Android Studio, помимо тех возможностей, которые предоставляет Intellij IDEA, дополнительные возможности (визуальные и функциональные), оказывающие поддержку при разработке. Встроенный SDK помогает оперативно избавляться от ошибок, связанных с некорректной версий API, причем студия сама предлагает нужную версию и её непосредственную установку.

Возможности интерфейса оказывают неизгладимую помощь при разработке. Здесь и возможности изменения отображения экрана, просмотр добавленных view элементов, отображение экрана на любом формате устройств.

Дизайн самой студии переработан, имеет ряд усовершенствований по сравнению с Intellij IDEA. Экраны логов, сообщений, свойств, занимают большую часть экрана и имеют возможность сворачивания в нижнюю панель, что очень удобно при разработке. Таким образом Android Studio лучше всего адаптирована для разработки под Android.

1.9. Выбор средства для взаимодействия с базой данных

В Java одной из возможностей работы с удаленной базой данных является использование технологии JDBC. JDBC представляет собой Java API, то есть это совокупность классов и интерфейсов. Данная технология является унифицированной и позволяет одинаково хорошо работать с различными типами СУБД путем использования полиморфизма через реализуемые интерфейсы.

JDBC API делится на несколько уровней и включает в себя: прикладной уровень, уровень драйвера, описательный уровень. Конкретная реализация представляет собой JDBC драйвер для работы с конкретной СУБД. Данная архитектура представлена на рисунке 2.3.

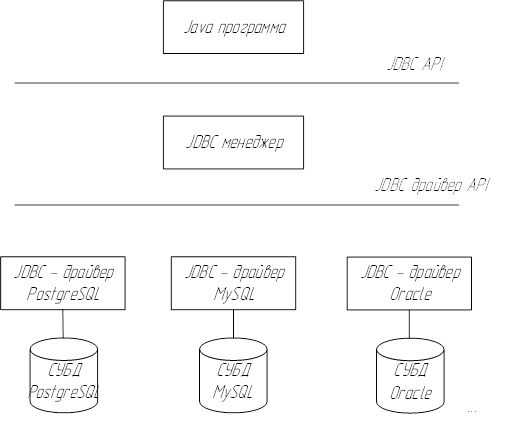


Рисунок 2.3. Архитектура JDBC

Выводы

Таким образом, исходя из требований к реализуемой системе, рассмотрения возможностей наиболее подходящих инструментов, вариантов разработки и последующего их сравнения было решено использовать следующие решения:

В качестве ОС для разработки были выбраны Windows и Android в силу имеющегося ряда преимуществ и учета конкретной целевой аудитории для которой она предназначается.

Исходя из требований к системе и характеристик каждой из рассмотренных СУБД в качестве оптимальной была выбрана MySQL.

В качестве инструментов для реализации серверной части было решено использовать: язык программирования C#, среду разработки Microsoft Visual Studio и технологию Windows Forms для реализации графического интерфейса.

Для разработки мобильного приложения было решено использовать следующие инструменты:

В качестве языка программирования был выбран язык программирования Java.

В качестве интегрированной среды разработки была выбрана адаптированная специально под ОС Android - Android Studio.

Для взаимодействия с удаленной базой данных было решено использовать технологию JDBC, поскольку данная технология является наиболее подходящей для решения поставленной задачи и к тому же проста и удобна в использовании.

2. ПРОЕКТИРОВАНИЕ КОМПОНЕНТОВ ПРОГРАММНОГО ПРОДУКТА

2.1. Разработка структуры системы

Разрабатываемая система состоит из следующих компонентов:

База данных – хранит всю необходимую информацию об имеющимся на предприятии оборудовании.

Серверная часть – представляет собой компьютер-сервер на базе ОС Windows с базой данных и приложением для добавления, изменения, получения, удаления информации об имеющимся на предприятии контрольно-измерительном оборудовании с возможностью вывода отчета в формате Excel.

Клиентская часть – мобильное приложение на базе ОС Android для получения информации об оборудовании из удаленной базой данных путем считывания его QR кода с возможностью генерации такого кода с последующим сохранением его в памяти телефона.

2.2. Разработка базы данных

База данных состоит из следующих таблиц:

* перечень оборудования;
* тип;
* тип-свойство;
* свойство;
* оборудование-свойство.

Структура базы данных представлена на рисунке 3.1.

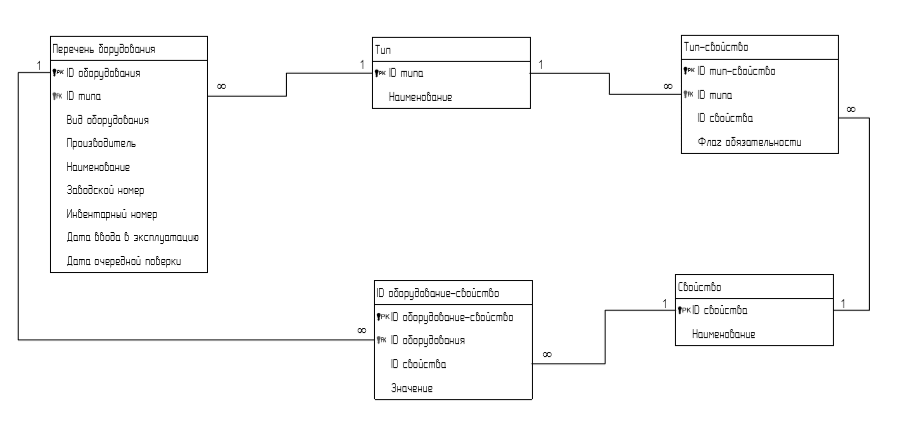


Рисунок 3.1. Структура базы данных

Ниже приведено описание каждой из таблиц.

Таблица 1 – «accounts» - хранит общую информацию о счетах клиента

| Наименование поля | Тип данных | Признак ключа | Примечание |
| --- | --- | --- | --- |
| id | int | PK | Уникальное значение |
| account\_number | string |  | Уникальное значение.  Хранит в себе номер счета клиента |
| user\_id |  | FK | Уникальное значение  Включает в себя id клиента |
| Вид оборудования | enum |  | Включает в себя: измерительное оборудование, индикаторный прибор, вспомогательное оборудование. |
| Производитель | varchar(100) |  |  |
| Наименование | varchar(100) |  |  |
| Заводской номер | varchar(50) |  | Служит в качестве зашифрованной информации в QR коде оборудования, поскольку однозначно определяет последнее. |
| Инвентарный номер | varchar(50) |  |  |
| Дата ввода в эксплуатацию | date |  |  |
| Дата очередной поверки | date |  |  |

Таблица 2 – «Тип» - справочник для хранения всех типов оборудования

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование поля | Тип данных | Признак ключа | Примечание |
| ID типа | int | PK | Уникальное значение |
| Наименование | varchar(50) |  | Наименование типа |

Таблица 3 – «Свойство» - справочник для хранения всех свойств оборудования

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование поля | Тип данных | Признак ключа | Примечание |
| ID свойства | int | PK | Уникальное значение |
| Наименование | varchar(50) |  | Наименование свойства |

Таблица 4 – «Тип-свойство» - служит для обеспечения связи «многие ко многим» между таблицами «Тип» и «Свойство»

| Наименование поля | Тип данных | Признак ключа | Примечание |
| --- | --- | --- | --- |
| ID тип-свойство | int | PK | Уникальное значение |
| ID типа | int | FK | Уникальное значение. Указывает на запись в таблице Тип. |
| ID свойства | int | FK | Уникальное значение. Указывает на запись в таблице Свойство. |
| Флаг обязательности | bit |  | Указывает на обязательность наличия свойства у данного типа оборудования |

Таблица 5 – «Оборудование-свойство» - служит для обеспечения связи «многие ко многим» между таблицами «Перечень оборудования» и «Свойство»

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование поля | Тип данных | Признак ключа | Примечание |
| ID оборудование - свойство | int | PK | Уникальное значение |
| ID оборудования | int | FK | Уникальное значение. Указывает на запись в таблице «Перечень оборудования». |
| ID свойства | int | FK | Уникальное значение Указывает на запись в таблице «Свойство». |
| Значение | varchar(500) |  | Содержит значение данного свойства. |

2.3. Построение диаграммы вариантов использования Windows приложения

Данное приложение позволяет помимо получения информации об оборудовании, производить различные действия над ним. Подробное описание возможностей данного компонента системы удобно представить при помощи языка прецедентов. Прецедент представляет собой один из вариантов использования приложения. Совокупность же прецедентов помогает определить перечень действий для достижения необходимого результата. Разработанная диаграмма представлена на рисунке 3.2.

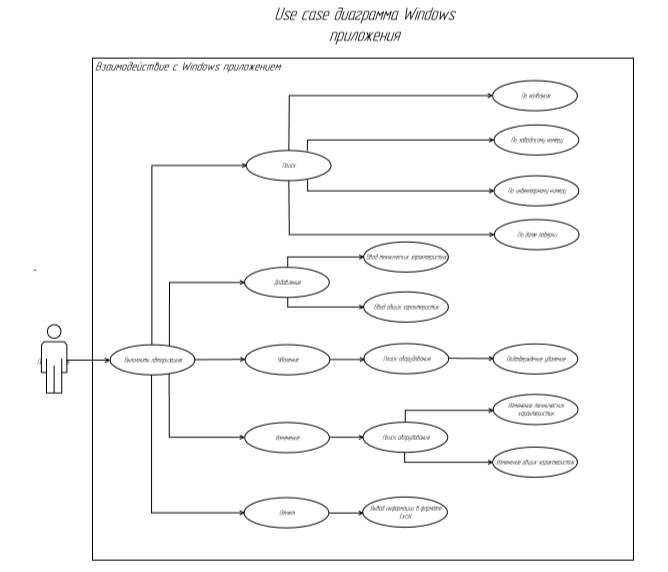


Рисунок 3.2. Use case диаграмма Windows приложения

2.4. Структура Android приложения

Мобильное приложение позволяет получать информацию о нужном оборудовании путем считывания его QR кода и при необходимости генерации изображения такого кода с последующим сохранением его в памяти телефона. Ниже приведена блох-схема иллюстрирующая алгоритм получения информации об оборудовании.



Блок-схема 1. Алгоритм получения информации об оборудовании

Данный компонент системы представляет собой Android приложение структура которого строится на совокупности описанных ниже модулей (активностей) и набора вспомогательных классов.

— Главная активность определена в файле MainActivity.java. Данная активность предназначена для вывода стартового экрана при запуске приложения, экрана авторизации пользователя. В случае успешного соединения происходит вызов следующей активности.

— Активность, описанная в файле AssigningActivity.java отвечает за окно выставления настроек для и вызов камеры с заданными параметрами.

— Активность, описанная в файле InformationList.java вызывается после считывания очередного QR-кода. Данная активность отвечает за получение информации из базы данных о приборе с соответствующим кодом и её выводе на экран мобильного устройства.

— Активность, описанная в файле BarcodeCaptureActivity.java вызывается после нажатия кнопки считывания QR-кода. Данная активность запускает превью камеры, выполняет обнаружение QR- кодов на данном превью, отображает их значения. При считывании кодов на каждый из них накладываются графические рамки, для определения позиции, идентификатора и размера. Все вышеперечисленные функции выполняются во вспомогательных классах, описанных далее.

— Вспомогательный класс ConnectionDB.java отвечает за подключение к удаленной базе данных.

— Класс CameraSource.java предоставляет функционирование камерой, для предварительного просмотра. Управляет заданными настройками, запускает камеру, обеспечивает поворот.

— Класс CameraSourcePreview.java отвечает за управление превью камеры. Её отображением, размерами, разрешением на запуск и т.д.

— Класс GraphicOverlay.java реализует отображение графических объектов поверх созданного превью. Поддерживает добавление, удаление, обновление элементов и управление масштабированием графического объекта относительно свойств превью камеры.

— Вспомогательный класс BarcodeTrackerFactory.java служит для реализации паттерна «Фабрика», для создания трекеров QR-кода. По одному трекеру на каждый код.

— Класс BarcodeGraphicTracker.java представляет трекер для обнаружения QR-кодов. Занимается отслеживанием кодов, наложением на них графики и удалением при необходимости.

— Класс BarcodeGraphic.java отвечает за отрисовку экземпляра накладываемого на код изображения с учетом его идентификатора, размера и положения.

Mobile Vision API

Для распознавания QR-кодов в приложении используется стандартная библиотека Mobile Vision API. Таким образом нет необходимости интегрировать сторонний API в проект, все что необходимо это правильно интегрировать службы Google Play. Включает в себя два пакета: для распознавания лиц и штрих-кодов. Последняя используется в данной разработке. Данная API позволяет распознавать штрих-коды в любом положении экрана (ориентации) и в режиме реального времени. Поддерживает распознавание не только QR-кода, но и штрих-коды других 1D и 2D форматов (EAN-13, Code-39, ITF, PDF-417, AZTEC и т.д.).

Взаимодействие основных классов и интерфейсов

Процесс распознавания реализуется взаимодействием между следующими классами и интерфейсами: CameraSourcePreview, CameraSource, BarcodeDetector, MultiProcessor, BarcodeTrackerFactory, BarcodeGraphicTracker, BarcodeGraphic.

Ниже представлен алгоритм распознавания в виде блок-схемы.



Блок-схема 2. Алгоритм распознавания штрих-кода

На первом этапе, для отслеживания штрих-кода, создается детектор штрих кода – экземпляр BarcodeDetector. Используя созданный детектор, создается связанный с ним конвейер процесса, для дальнейшего получения результатов обнаружения. Мультипроцессор служит для создания трекера (BarcodeGraphicTracker <BarcodeGraphic>), для каждого штрих-кода, оказавшегося в зоне видимости камеры. BarcodeGraphic накладывает на изображение с камеры, ограничивающий прямоугольник и значение для каждого результата штрих-кода.

MultiProcessor – это компонент для работы с произвольным количеством обнаруженных элементов (штрих-кодами), т.е. позволяет одновременно сканировать и обнаруживать несколько штрих-кодов. По мере движения камеры, обновления для штрих-кодов направляются в каждый из соответствующих экземпляров BarcodeGraphicTracker, которые по средством BarcodeGraphic осуществляют перерисовку конкретного штрих-кода. Когда штрих-код исчезает из зоны видимости, многопроцессор уничтожает связанный с ним экземпляр BarcodeGraphicTracker. Таким образом осуществляется динамическое создание / отслеживание / уничтожение индивидуального трекера для каждого кода.

Для отображения изображения с камеры, служит CameraSourcePreview. CameraSourcePreview отвечает за пользовательский интерфейс предварительного просмотра. CameraSource управляет получением изображений с CameraSourcePreview и передачей их детектору для последующей обработки.

Ниже приведена диаграмма описанного взаимодействия (рисунок 3.3).

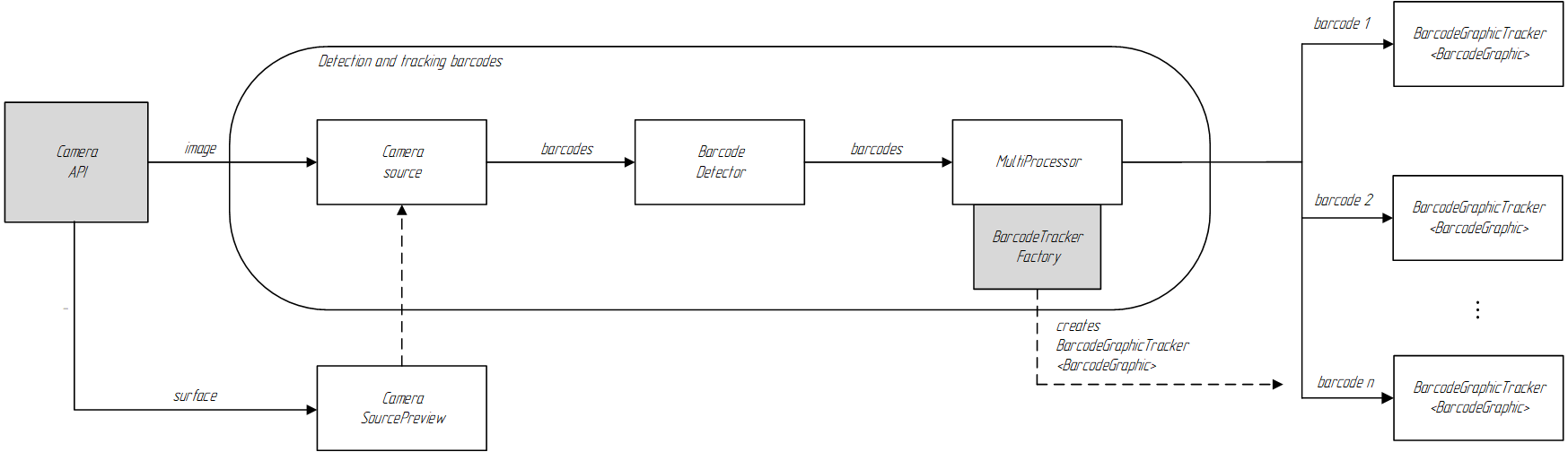


Рисунок 3.3. Диаграмма зависимости основных классов и интерфейсов

2.5. Организация сетевого взаимодействия.

В разрабатываемой системе клиенты - мобильные устройства с приложением, формирующим запросы к серверу на получение информации о конкретном оборудовании для последующего отображения этой самой информации на экране. Ниже приведена блок-схема иллюстрирующая алгоритм подключения клиента к серверу базы данных.



Блок-схема 3. Алгоритм подключения к серверу БД

На транспортно-сетевом уровне связь между серверами и клиентами осуществляется с использованием протокола TCP/IP. Поверх него, для обмена информацией, функционирует собственный бинарный клиент-серверный протокол mysql. Разграничение прав доступа реализуется средствами mysql.

Описанный стек протоколов представлен на рисунке 3.4.

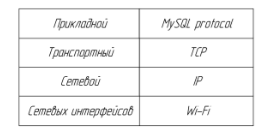


Рисунок 3.4. Стек протоколов разрабатываемой системы

**Использование JDBC**

Для организации сетевого взаимодействия сначала необходимо подключить драйвер JDBC для работы Android приложения с конкретным типом базы данных, в нашем случае с MySQL. Для организации такой работы необходимо:

— Произвести загрузку и регистрацию необходимого класса драйвера (jar библиотека).

— Указать путь к драйверу (classpath).

Основными интерфейсами и классами для работы через JDBC служат: Connection, Statement, ResultSet, DriverManager.

Класс DriverManager является посредником между программой и драйвером. Является основным интерфейсом JDBC. Основная функция – присоединить Java приложение к требуемому драйверу, для конкретной СУБД.

Connection служит для подключения к БД. Результатом успешного подключения служит созданный объект, описывающий данное соединение. Программа получает ссылку на этот объект для дальнейшей работы. Этот объект содержит характеристики состояние соединения с базой данных, и кроме того предоставляет средства для контроля транзакций.

Объекты классов, реализующих интерфейс Statement служат для создания и выполнения SQL запросов к БД. Служит контейнером для SQL выражений, включающих не только текст запроса, но и параметры выражения, его состояние.

Объекты классов, реализующих интерфейс ResultSet используются для получение результатов запросов, представляют собой коллекции строк.

Ниже приведена диаграмма взаимодействия между элементами JDBC, приложением и БД (рисунок 3.5).

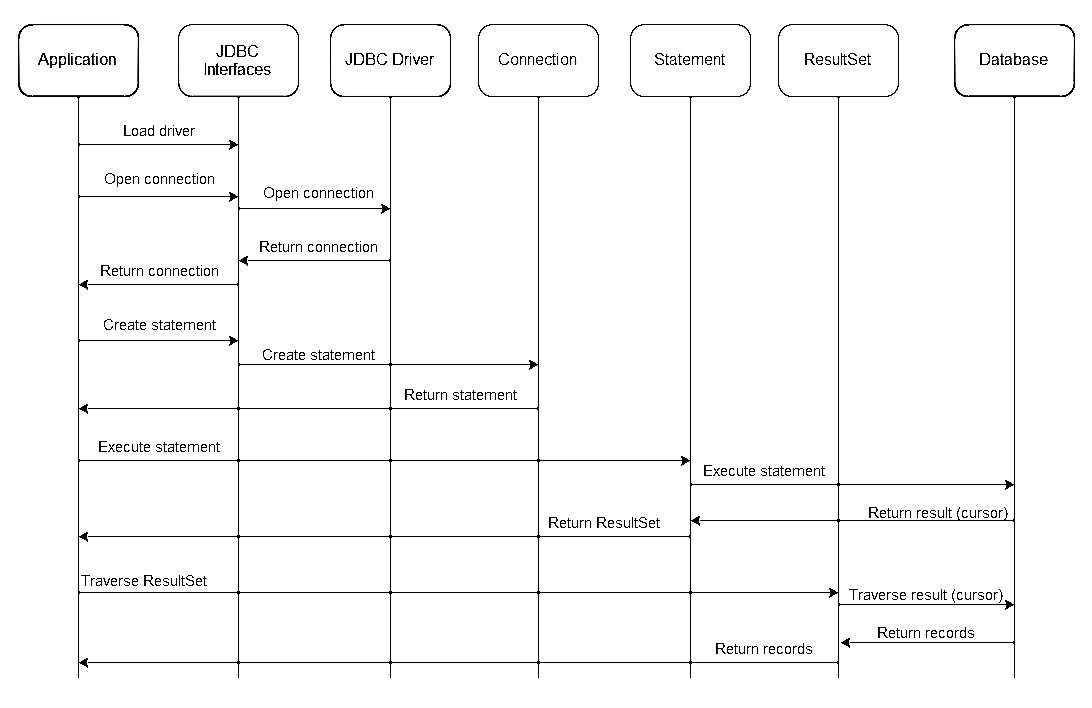


Рисунок 3.5. Взаимодействие между компонентами JDBC, приложением и БД.

Выводы

В результате были выполнены проектирование и реализация компонентов программного продукта: базы данных, серверной части и клиентской части. Полность учтены функциональные требования технического задания.

3. ТЕСТИРОВАНИЕ И ИНТЕГРАЦИЯ КОМПОНЕНТОВ ПРОГРАММНОГО ПРОДУКТА

3.1. Тестирование системы

Тестирование системы производилось с использованием персонального компьютера в качестве сервера и мобильных устройств на базе ОС Android. На сервере тестировалась работа Windows приложения по учету оборудования и включало в себя проверку всех возможных вариантов использования с последующим выводом отчета. На мобильных устройствах производилось тестирование возможностей клиентского приложения, а также получение информации из базы данных с нескольких устройств одновременно.

Требования к Windows приложению

Для работы Windows приложения необходимо, чтобы компьютер обладал следующими минимальными характеристиками:

— процессор с тактовой частотой не менее 1 ГГц;

— ОЗУ 1 Гб;

— 2 Гбайт свободного пространства на диске;

— Microsoft Windows 7/8/8.1/10

— MySQL Server

— .NET Framework 4.5.2 или выше

Для взаимодействия с приложением необходимо наличие мыши или монитора.

Требования к мобильному приложению

Минимальные требования для работы мобильного приложения:

— мобильное устройство под управлением ОС Android версии 4.0 или выше;

— 9,6 Мб свободного пространства;

— наличие камеры;

Разрешения:

— просмотр сетевых подключений;

— доступ к камере;

— съемка фото;

— управление функциями вспышки и автофокуса.

Требования к программному обеспечению

Для работы приложения на сервере-компьютере с базой данных необходимо установить и настроить MySQL сервер.

3.2. Руководство администратора

Настройка доступа к базе данных

Для работы необходимо установить и настроить mysql сервер. Подробное руководство изложено в соответствующей документации <https://dev.mysql.com/doc/refman/8.0/en/windows-installation.html>. Для обеспечения доступа к удаленной базе данных сервер должен быть запущен. На уровне сервера необходимо определить уровень доступа пользователей к базе данных сервера. После добавления пользователей и ввода соответствующих разрешений, система безопасности MySQL будет самостоятельно осуществлять аутентификацию очередного пользователя при попытке доступа.

Для добавления пользователя и настройки для него определенных прав доступа необходимо выполнить следующие действия:

Добавить нового пользователя:

CREATE USER 'имя нового пользователя'@'адрес устройства' IDENTIFIED BY 'пароль';

Для того, чтобы пользователь мог подключаться к базе данных с любого устройства, необходимо в качестве адреса указать символ %.

Установить привилегии для пользователя:

GRANT [тип прав] ON [имя базы данных].[имя таблицы] TO ‘имя пользователя’@'адрес устройства;

Доступные привилегии:

* CREATE – Разрешить создание базы данных/таблиц
* SELECT – Разрешить производить выборку данных
* INSERT – Разрешить добавление новых записей в таблицы
* UPDATE – Разрешить изменение существующих записей в таблицах
* DELETE – Разрешить удалять записи из таблиц
* DROP – Разрешить удалять записи в базе данных/таблицах

Для того, чтоб изменения вошли в силу запустите команду обновления: FLUSH PRIVILEGES;

3.3. Руководство пользователя

Руководство к Windows приложению

Для запуска приложения необходимо запустить на выполнение файл. «AccountingSystemCMequipment.exe». На экране появится окно авторизации (рисунок 4.1).

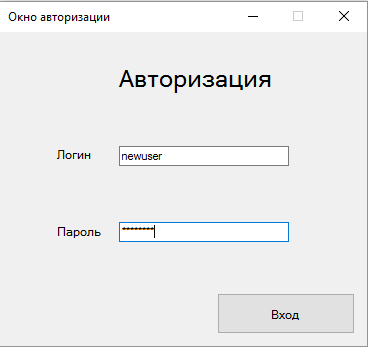


Рисунок 4.1. Окно авторизации

Если авторизация прошла успешно на экране появится окно главного меню (рисунок 4.2).

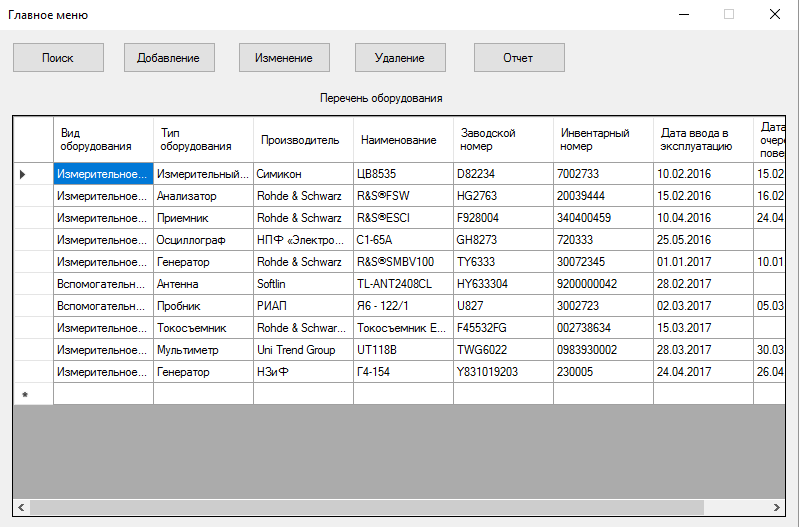


Рисунок 4.2. Главное меню

Кнопка «Поиск» позволяет осуществить поиск оборудования. При желании можно указать критерии поиска (рисунок 4.3).

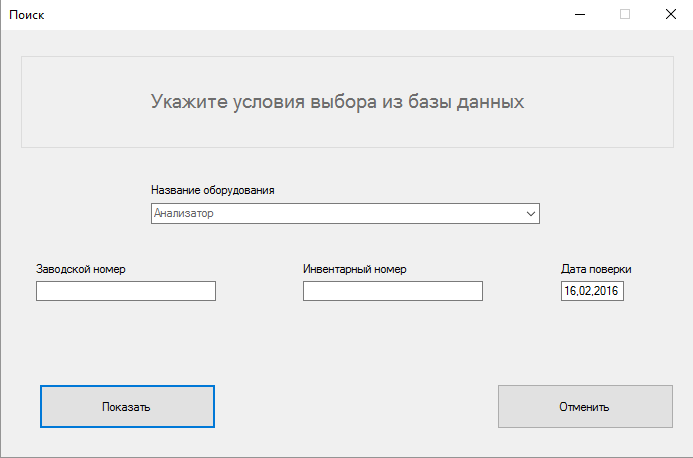


Рисунок 4.3. Окно «Поиск»

Рисунок 4.4 иллюстрирует результат выполнения поиска оборудования.

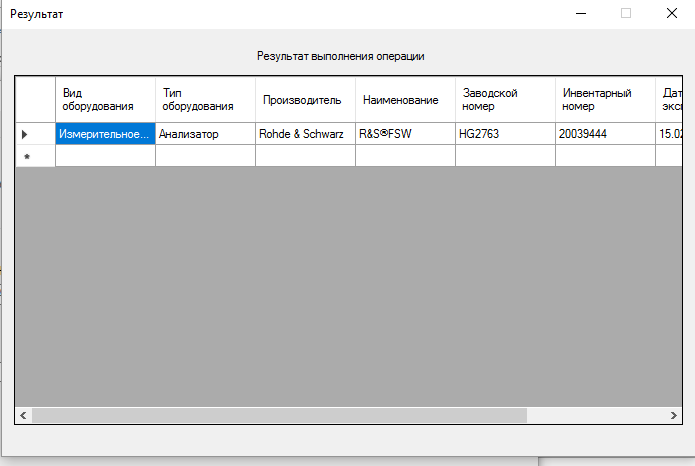


Рисунок 4.4. Результат выполнения операции поиска

Для отображения технических характеристик, необходимо нажать в любом месте строки выбранного оборудования. Появится окно с имеющийся информацией о данном оборудовании (рисунок 4.5).

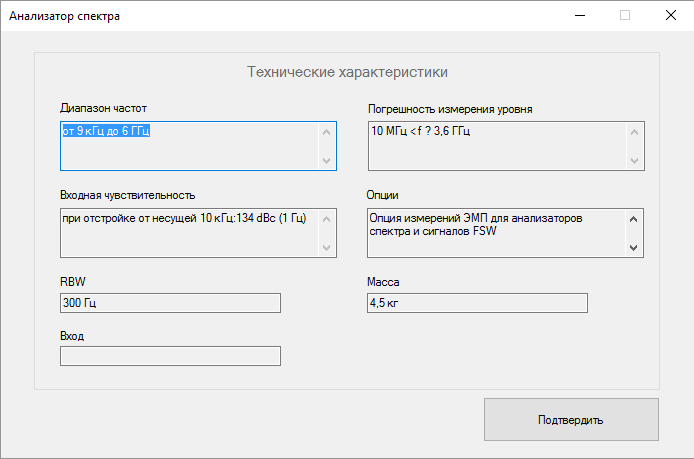


Рисунок 4.5. Технические характеристики выбранного оборудования

Для обеспечения удобства и возможности сравнения можно производить поиск и открывать несколько окон с результатами одновременно.

Кнопка «Добавление» позволяет добавить оборудование в базу данных (рисунок 4.6).

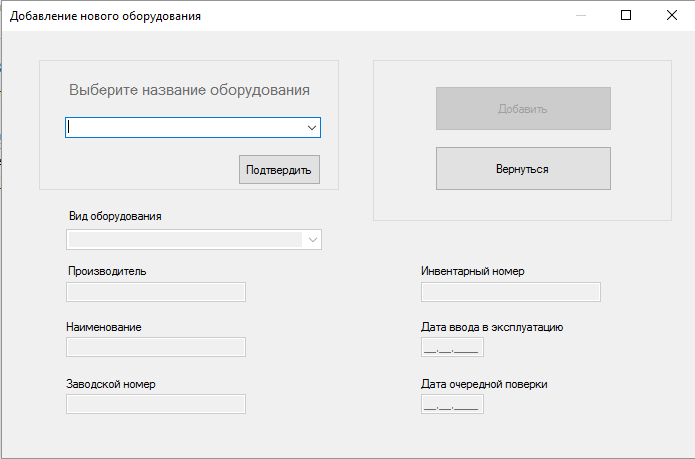


Рисунок 4.6. Добавление нового оборудования

Сначала необходимо выбрать тип оборудования. Выбор определит окно, соответствующее данному типа для ввода технических характеристик (рисунок 4.7).

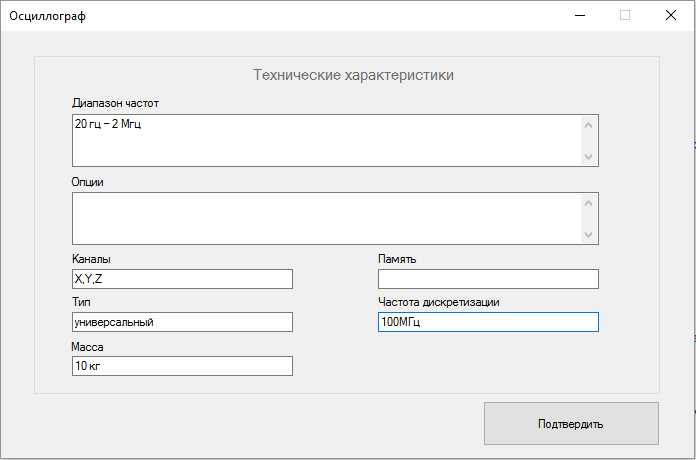


Рисунок 4.7. Окно ввода технических характеристик для осциллографа

После подтверждения введенных данных можно приступить к вводу общих характеристик (рисунок 4.8).

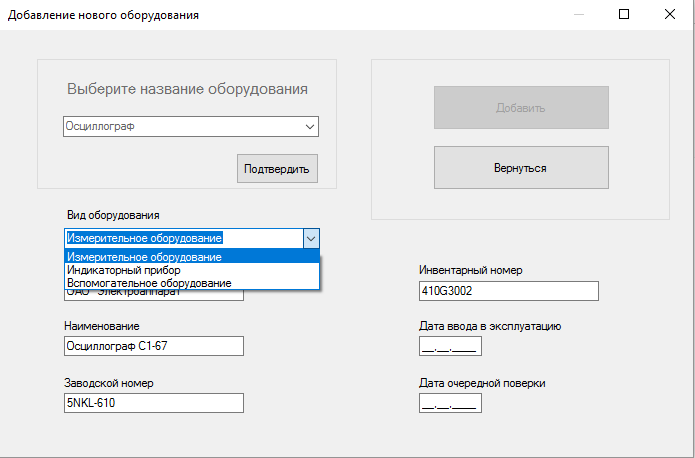


Рисунок 4.8. Ввод основных характеристик

В случае ввода некорректных данных появится значок, сообщающий об ошибке, рядом с соответствующем полем (рисунок 4.9).

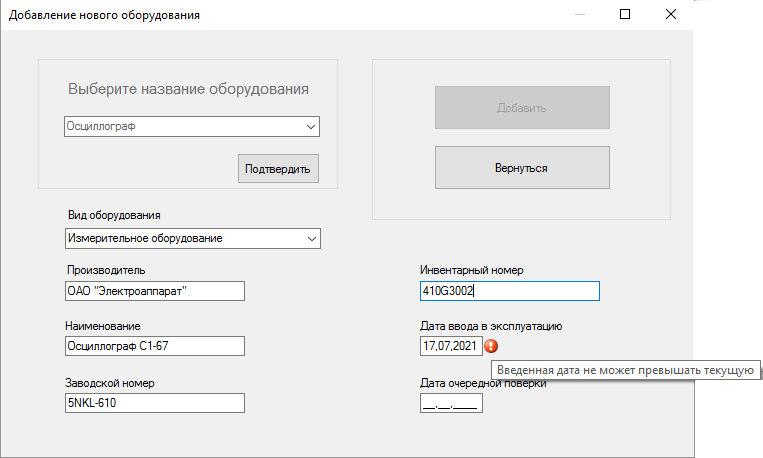


Рисунок 4.9. Уведомление об ошибке

Кнопка «Добавить» будет не доступна до тех пор, пока не будут корректно заполнены все обязательные поля.

Кнопка «Изменение» позволяет изменить характеристики выбранного оборудования (рисунок 4.10).

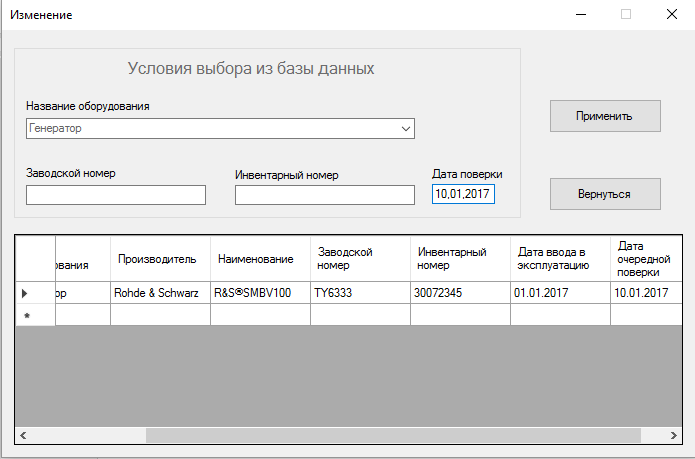


Рисунок 4.10. Окно «Изменение» с указанием критериев

При желании можно указать критерии выбора оборудования из базы данных.

Для отображения характеристик, необходимо нажать в любом месте строки выбранного оборудования. Появится окно «Общие сведения» в котором можно вносить желаемые изменения (рисунок 4.11).

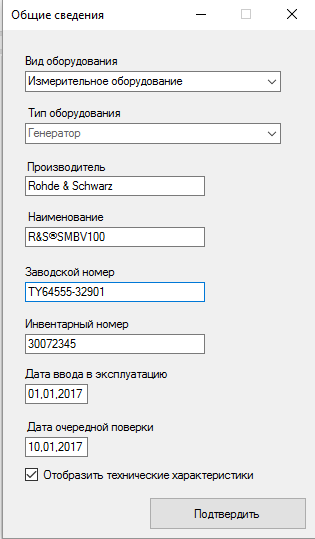


Рисунок 4.11. Окно изменения общих характеристик

Для изменения технических характеристик необходимо поставить галочку рядом с надписью «Отобразить технические характеристики» (рисунок 4.12).

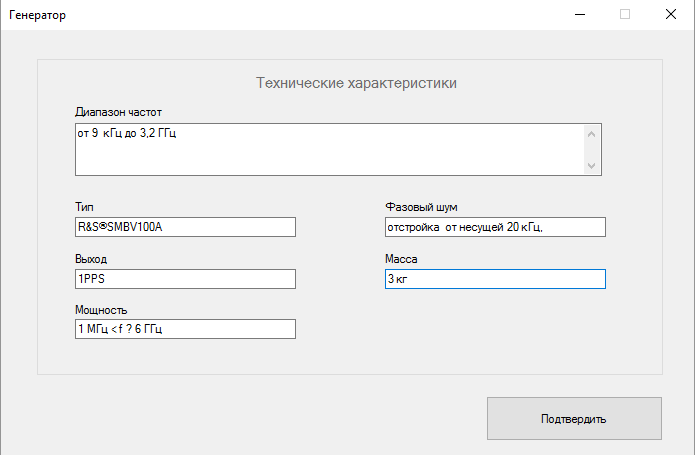


Рисунок 4.12. Окно изменения технических характеристик для оборудования выбранного типа

Кнопка «Удаление» позволяет удалить выбранное оборудование.

Для выбора оборудования для удаления, необходимо нажать в любом месте строки выбранного оборудования. Появится окно «Подтверждения», в котором необходимо подтвердить или отклонить удаление (рисунок 4.13).

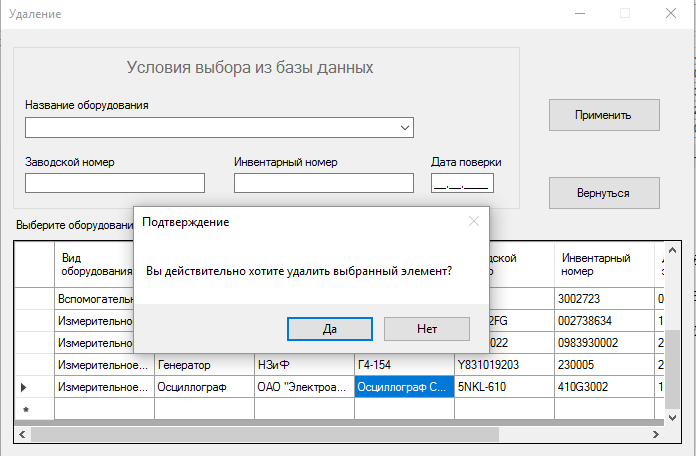


Рисунок 4.13. Окно «Подтверждение»

Кнопка «Отчет» позволяет вывести отчет, содержащий все таблицы базы данных в формате Excel (рисунок 4.14).

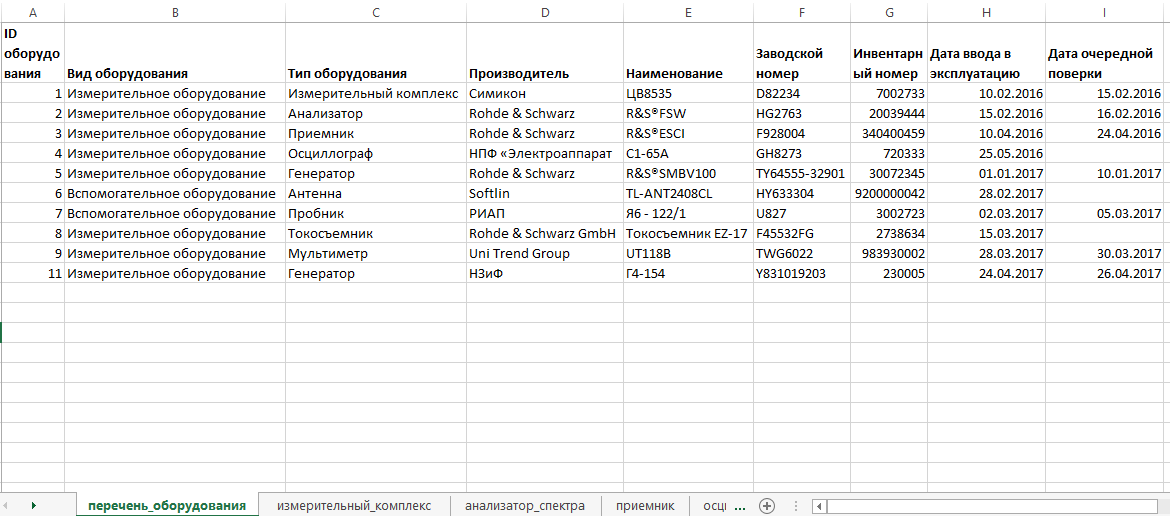


Рисунок 4.14. Отчет в формате Excel **-** Перечень оборудования.

Руководство к Android приложению

Для запуска приложения необходимо запустить исполняемый файл «BarcodeReader.apk», после чего отобразиться главное окно приложения. В данном окне необходимо ввести данные для подключения: логин и пароль (рисунок 4.15). Адрес подключения указывается в настройках приложения (рисунок 4.16).



Рисунок 4.15. Окно авторизации

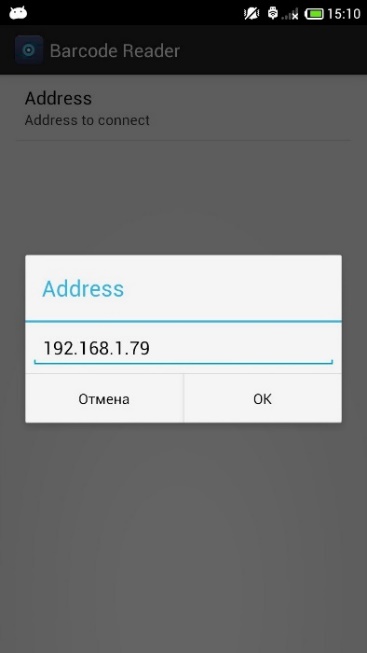


Рисунок 4.16. Ввод адреса подключения

В случае некорректного ввода данных для подключения или отсутствия последнего, внизу экрана появится уведомление о ошибке.

В случае успешного подключения, на экране отобразиться окно главного меню (рисунок 4.17).



Рисунок 4.17. Окно главного меню

В случае выбора пункта сканирования кода, на экране отобразится окно для предварительного выставления настроек камеры. Имеется возможность включения использования автофокуса и/или вспышки (рисунок 4.18).



Рисунок 4.18. Окно указания настроек камеры

Нажатие клавиши «Read Barcode» активирует запуск камеры с указанными настройками для считывания QR-кода. Распознаваемые штрих-коды будут выделяться рамкой. Снизу рамки, области считывания, будут появляться данные о получаемой информации из кода (рисунок 4.19).



Рисунок 4.19. Процесс считывания QR-кода

Полученная информация об оборудовании отобразиться на следующем экране (рисунок 4.20).



Рисунок 4.20. Полученная информация из базы данных

В случае, если на сервере базы данных не содержится информация о данном оборудовании, пользователь получит уведомление об отсутствии таких данных (рисунок 4.21).



Рисунок 4.21. Уведомление об отсутствии данных

При выборе в главном меню пункта генерации кода на экране появится соответствующее окно. В строке генерации необходимо ввести заводской номер оборудования и нажать клавишу «Generate». Сгенерированное изображение появится в центральной части экрана, а в нижней части экрана будет указан путь сохранения указанного изображения (рисунок 4.22).



Рисунок 4.22. Окно генерации кода

Выводы

В результате были выполнены тестирование компонентов программного продукта. Разработана техническая документация.

ЗАКЛЮЧЕНИЕ

В результате выполнения выпускной квалификационной работы была создана система, позволяющая вести автоматизированный учет контрольно-измерительного оборудования в подразделении предприятия. Результат соответствует всем заранее определенным требованиям технического задания. Разработанная система обладает гибкостью, многофункциональностью и удобством в эксплуатации. Внедрение данной системы на предприятие позволит увеличить производительность труда за счет качественного анализа, планирования имеющихся ресурсов и уменьшения времени поиска необходимой информации, а также будет приносить экономическую выгоду в силу отсутствия затрат на материалы для производства бумажной документации.

Поскольку поставленная задача была выполнена в полном объеме дальнейшее её расширение не планируется. Однако разработанная архитектура позволяет при необходимости улучшить и расширить данную систему.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Основная литература

1. Романенко, В.В. Объектно-ориентированное программирование : учебное пособие / В.В. Романенко ; Министерство образования и науки Российской Федерации, Томский Государственный Университет Систем Управления и Радиоэлектроники (ТУСУР). - Томск : Томский государственный университет систем управления и радиоэлектроники, 2014. - 475 с. : ил. - Библиогр.: с. 442. ; То же [Электронный ресурс]. - URL: <http://biblioclub.ru/index.php?page=book&id=480517>.
2. Николаев, Е.И. Объектно-ориентированное программирование : учебное пособие / Е.И. Николаев ; Министерство образования и науки Российской Федерации, Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Северо-Кавказский федеральный университет». - Ставрополь : СКФУ, 2015. - 225 с. : ил. - Библиогр. в кн. ; То же [Электронный ресурс]. - URL: <http://biblioclub.ru/index.php?page=book&id=458133>.
3. Суханов, М.В. Основы Microsoft .NET Framework и языка программирования C# : учебное пособие / М.В. Суханов, И.В. Бачурин, И.С. Майоров ; Министерство образования и науки Российской Федерации, Федеральное государственное автономное образовательное учреждение высшего профессионального образования Северный (Арктический) федеральный университет им. М.В. Ломоносова. - Архангельск : ИД САФУ, 2014. - 97 с. : схем., табл., ил. - Библиогр. в кн. - ISBN 978-5-261-00934-4 ; То же [Электронный ресурс]. - URL: <http://biblioclub.ru/index.php?page=book&id=312313>.

Дополнительная литература

1. Мейер, Б. Объектно-ориентированное программирование и программная инженерия [Электронный ресурс] / Б. Мейер. — 3-е изд. — Электрон. текстовые данные. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Эр Медиа, 2019. — 285 c. — 978-5-4486-0513-0. — Режим доступа: <http://www.iprbookshop.ru/79706.html>
2. **добавить свои источники, общим числом не менее 10**