**Введение**

В современном мире трудно представить жизнь без IT-технологий. В этой цифровой эпохе использование программного обеспечения становится неотъемлемой частью нашей реальности. Все ищут способы попасть в IT, выучить язык программирования и вложить свой вклад в развитие технологий. Это инструмент, который облегчает обучение, улучшает процессы обучения и управления информацией, а также способствует развитию профессиональных навыков.

Важность создания программного средства трудно переоценить. Этот инструмент не только обеспечит удобное управление учебным материалом и выполнение практических заданий, но и предоставит пользователю возможность эффективной работы с информацией и обмена опытом с коллегами.

Целью разработки курсового проекта на тему «Создание программного средства по учебному предмету Инструментальное программное обеспечение» является реализация инструмента, который позволит студентам учиться эффективно, проверять свои знания и обмениваться опытом с коллегами, благодаря удобному интерфейсу, возможности выполнять практические задания, формированию отчетов, изучению и пометок в документации.

Актуальность программного средства заключается в том, что оно отвечает современным требованиям к образованию и управлению информацией, предоставляя пользователю инновационные инструменты для обучения и развития профессиональных навыков.

В первом разделе «Описание задачи» основное внимание уделяется выявлению ключевых задач, необходимых для успешной реализации проекта. Это включает в себя выявление основных функциональных требований к программному средству, а также план разработки.

Во втором разделе «Проектирование программного средства» представлено описание структуры программного средства, включая создание лаконичного интерфейса с использованием меню, кнопочных форм и панелей инструментов, а также организацию иерархии классов и работы с информацией.

В третьем разделе «Реализация программного средства» затронуты технические аспекты создания программного средства, включая выбор используемых технологий, алгоритмы и архитектуру приложения, а также реализацию всех функций и возможностей, описанных в задании.

Четвертый раздел «Применение» посвящен обзору возможностей и сфер применения разработанного программного средства, а также его влиянию на повседневную деятельность пользователей, обучение и управление информацией.

В Заключении проведен анализ результатов разработки, выделены основные достижения и определены перспективы дальнейшего развития проекта.

# Описание задачи

## Анализ предметной области

Целью разработки курсового проектирования на тему «Создание программного средства по учебному предмету Инструментальное программное обеспечение» является реализация программного средства, которое предоставляет пользователям возможность входа в личный аккаунт или создания нового, просмотра доступных учебных материалов, регистрации в приложении, добавления материалов в избранное и получения краткого описания учебных курсов.

Предметная область включает разработку программного средства, предназначенного для упрощения процесса доступа к учебным ресурсам. В контексте использования данного приложения пользователи могут записываться на различные курсы, использовать возможности приложения для получения точной информации и прохождения курсов. Приложение должно обеспечивать удобное взаимодействие с предлагаемыми учебными материалами, предоставлять возможность выбора удобного времени для участия, а также обеспечивать обратную связь с преподавателями для уточнения нужной информации.

Компьютерная обработка значительно упрощает процесс доступа к учебным материалам по сравнению с реальным визитом. Она повышает спрос пользователей и их заинтересованность, обеспечивает быстрый доступ к доступным курсам и их стоимости.

Потенциальными пользователями программного средства являются люди разных возрастов, поскольку учебные курсы востребованы как среди молодежи, так и среди более возрастных групп. Программное средство разработано с целью охвата максимальной клиентской базы и упрощения использования.

## Постановка задачи

Задачей проекта является разработка программного средства для просмотра и прохождения учебных курсов. Программа обеспечивает просмотр и ознакомление с курсами, их прохождения, просмотр видеороликов и прохождения практических заданий.

Программное средство будет включать в себя функционал для поиска по названию курса, что позволяет пользователям ориентироваться в большом объеме данных. Также будет возможность сохранения и просмотра пройденных курсов.

На рисунке 1.1 представлен аналог разрабатываемого программного средства «Stepik».

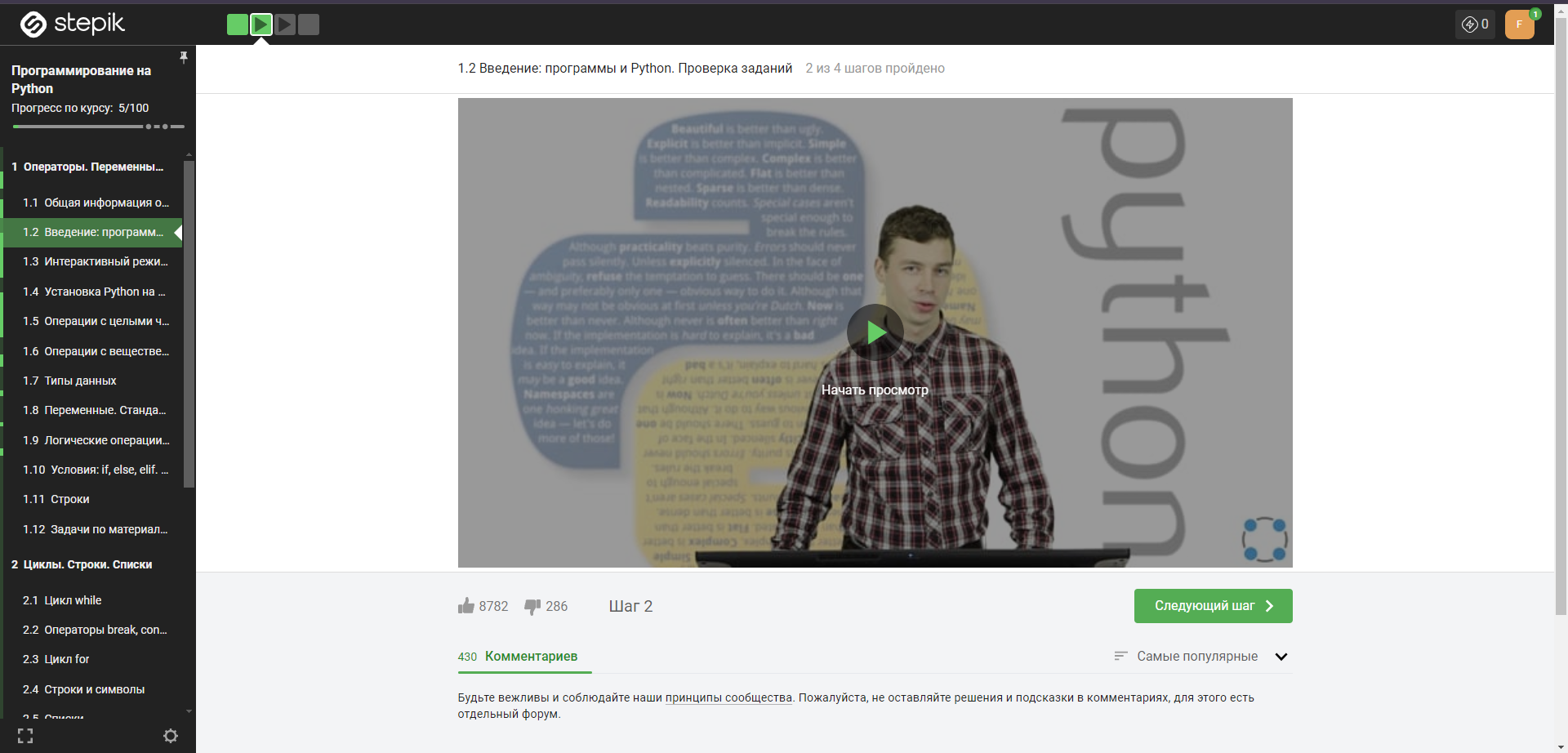


Рисунок 1.1 «Stepik»

Достоинства аналога:

1. Интерфейс: Простой и интуитивно понятный интерфейс облегчает навигацию и использование приложения.

2. Доступность: Возможность регистрации в приложении в любое время и из любого места, используя персональный компьютер

3. История прохождения курсов и отчеты: Хранение истории пройденных курсов, возможность создания отчетов

Недостатки аналога:

1. Зависимость от интернета: Необходимость постоянного подключения к интернету может ограничивать использование приложения в условиях слабого или отсутствующего интернета.

2. Технические проблемы: Возможные ошибки и сбои в работе приложения могут вызвать неудобства и снизить доверие пользователей.

3. Актуальность информации: Необходимость постоянного обновления информации о доступных курсах и их расписании может потребовать значительных усилий от организаторов.

Разрабатываемое программное средство будет включать в себя различные особенности, позволяющие более комфортно взаимодействовать с приложением.

# Проектирование системы

## Требования к приложению

Безопасность в приложении обеспечена на высоком уровне односторонним шифрованием в хэш и записью в бинарные файлы (.bin). В такие файлы записываются логин, пароль, личные данные пользователя из полей. В приложении реализована авторизация и регистрация, она локальная, потому что приложение поддерживается никакими онлайн серверами и является полностью локальным, что еще больше минимизирует риски взломов.

Интерфейс лаконичен и интуитивно понятен, он не требует от пользователя специальных технических знаний; все элементы управления расположены логично и последовательно. Реализована система вкладок в основной форме, чтобы не нагружать пользователя новыми дополнительными окнами.

Для удобства работы пользователей размеры кнопок и полей ввода, используется современный, читаемый шрифт, Century Gothic размером от 8 до 25 пунктов. Цветовая схема интерфейса темно-синего оттенка, что уменьшает нагрузку на глаза при длительной работе. Применяются сторонние элементы управления из специальных фреймворков, такие как закругленные, анимированные кнопки, выпадающие закругленные списки и закругленные поля ввода, соответствующие принципам удобства использования и красоты.

Минимальные требования к аппаратным и операционным ресурсам:

* OC: Windows 10
* Процессор: Pentium® 4 1.5 GHz / Athlon® XP
* Оперативная память: 2048 MB ОЗУ
* Видеокарта: Встроенная графика или любая дискретная видеокарта.

## Проектирование модели

Для отображения функциональной структуры объектов программного средства необходимо воспользоваться унифицированным языком моделирования UML, а именно диаграмму классов и диаграмму работы алгоритмов.

Диаграмма работы алгоритмов — это структурная диаграмма в языке UML является графическим представлением последовательности шагов выполнения алгоритма. Она помогает визуализировать логику алгоритма и его структуру, делая процесс понятным и легко читаемым. Данная диаграмма полезна при проектировании, отладке и понимании алгоритмов. Они помогают программистам и разработчикам лучше организовать свои мысли, выделить ключевые шаги и легко обнаружить ошибки или улучшить производительность алгоритма. Диаграмма работы алгоритмов представлена в графической части на листе 1.

Диаграмма классов — это структурная диаграмма в языке UML, которая отображает структуру системы, описывая классы, их атрибуты, методы и отношения между ними. Диаграмма классов помогает визуализировать архитектуру программного обеспечения и отношения между объектами. Данная диаграмма помогает разработчикам лучше понять структуру системы, улучшить проектирование программного обеспечения и облегчить коммуникацию между участниками проекта. Они являются важным инструментом для проектирования и документирования программных систем. Диаграмма классов представлена в графической части на листе 2.

## Концептуальный прототип

Концептуальный прототип – это общее видение продукта, его функций, логики взаимодействия с другими компонентами. Цель концептуального прототипа — получить обратную связь от пользователей и заинтересованных сторон на ранних этапах разработки, что позволяет уточнить требования и улучшить дизайн до начала полноценной разработки.

Окно авторизации включает в себя список кнопок, которые выполняют различные действия. Графическое представление прототипа окна авторизации представлено на рисунке 2.1. Кнопка регистрация оставляет пользователя на этой форме, добавляет кнопку зарегистрироваться и textBox для подтверждения пароля. Прототип представлен на рисунке 2.2.

Кнопка вход перебрасывает пользователя на рабочее окно приложения. Прототип рабочего окна приложения представлен на рисунке 2.3. После нажатия кнопки регистрации идет перенаправление на окно заполнения профиля. Где он должен подтвердить свой email и указать личные данные. Прототип предоставлен на рисунке 2.4

После заполнения профиля пользователя попросят удобно настроить для него приложение. Прототип окна настроек предоставлен на рисунке 2.5

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.1 – прототип окна авторизации

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.2 – окно регистрации

Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.3 – прототип рабочего окна приложения

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2.4 – прототип окна заполнения профиля

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описаниеРисунок 2.5 – окно настроек

# Описание реализации программного средства

## Инструменты разработки и применяемые технологии

Язык программирования C++ является мощным и гибким инструментом, который широко используется для разработки высокопроизводительных приложений, включая сетевые приложения. CLI (Common Language Infrastructure) предоставляет среду выполнения для управляемых приложений, позволяя использовать C++ для создания приложений .NET.

Инструмент разработки Visual Studio обеспечивает удобную среду для создания приложений на C++, включая поддержку Windows Forms для разработки графического интерфейса пользователя (GUI). Windows Forms позволяют создавать интерактивные приложения с использованием стандартных элементов управления.

Примеры приложений, которые могут быть разработаны с использованием C++, CLI, Windows Forms и Visual Studio, включают в себя клиент-серверные приложения, игры, системы управления базами данных и другие приложения, требующие высокой производительности и графического интерфейса.

В проектируемом программном системе (ПС) эта технология используется для создания клиентской части, которая будет взаимодействовать интернетом для получения данных о криптовалюте. Windows Forms обеспечит удобный способ создания графического интерфейса для пользователя, а C++ позволит обеспечить высокую производительность и эффективное управление ресурсами.

## 3.2 Функции: логическая и физическая организация

Реализация системы регистрации. Эта функция выполняется на стадии регистрации логина и пароля, она проверяет поля на корректность, в случае если какое-либо поле некорректно, она завершает свою работу. В ином случае функция вызывает другую, которая записывает в файл данные.

Void auth::buttonRegister\_Click(Object^ sender, EventArgs^ e) {

// Проверка на пустые поля

if (textBoxLogin1->Text == "") {

textBoxLogin1->BorderColor = Color::Red;

MessageWarning->Show("Введите логин");

return;

}

if (textBoxPassword1->Text == "" || textBoxPassword\_1->Text == "") {

textBoxPassword1->BorderColor = Color::Red;

textBoxPassword\_1->BorderColor = Color::Red;

MessageWarning->Show("Введите пароль");

return;

}

// Проверка на совпадение паролей

if (textBoxPassword1->Text != textBoxPassword\_1->Text) {

textBoxPassword1->BorderColor = Color::Red;

textBoxPassword\_1->BorderColor = Color::Red;

MessageError->Show("Пароли не совпадают", "Ошибка");

return;

}

// Проверка на соглашение

if (checkBoxTerms->Checked == false) {

MessageError->Caption = "Предупреждение";

MessageError->Text = "Для регистрации вам нужно прочесть условия использования";

MessageError->Show();

return;

}

// Проверка на существование уже такого пользователя

if (Directory::Exists(textBoxLogin1->Text)) {

MessageError->Caption = "Предупреждение";

MessageError->Text = "Пользователь под таким логином уже существует";

MessageError->Show();

return;

}

// Регистрация в случае если все условия соблюдены

panelMain->Visible = false;

Register();

}

Реализация функции Register(). Эта функция сохраняет логин, пароль пользователя.

Void auth::Register() {

// Получаем логин и пароль пользователя

String^ cliLogin = textBoxLogin1->Text;

String^ cliPassword = textBoxPassword1->Text;

// При получении данных меняем данные формы

USER = cliLogin;

labelWelcome->Text += cliLogin;

SetCenter(pageWelcome, labelWelcome, 1);

std::string Login = msclr::interop::marshal\_as<std::string>(cliLogin);

std::string Password = msclr::interop::marshal\_as<std::string>(cliPassword);

// Создаем папку с полученным логином

String^ folderPath = cliLogin;

Directory::CreateDirectory(folderPath);

// Путь к файлу Password - data.bin

String^ filePath = Path::Combine(folderPath, "data.bin");

// Хэшируем пароль

std::hash<std::string> hasher;

size\_t hashPassword = hasher(Password);

// Открываем файл для записи в бинарном режиме

FileStream^ fileStream = gcnew FileStream(filePath, FileMode::Create, FileAccess::Write);

BinaryWriter^ binaryWriter = gcnew BinaryWriter(fileStream);

binaryWriter->Write(hashPassword);

binaryWriter->Close();

fileStream->Close();

// Анимация, переход на следующую страницу.

animation\_mode = 1;

sec = 4;

timerTransition->Start();

}

Реализация функции входа в приложение:

Обработчик события кнопки «Войти» проверяет корректность введенных данных в поля, активировал ли пользователь функцию запоминания данных при последующих входах, и вызывает функцию входа типа Boolean, которая проверяет данные сохранённые в файлы и введенные пользователем. В случае сходства данных функция возвращает true иначе false.

Если функция возвращает true, обработчик активирует анимацию перехода на основную форму. Приветствие показывается если пользователь включил это в настройках.

Void auth::buttonComeIn\_Click(Object^ sender, EventArgs^ e) {

if (textBoxLogin->Text == "") {

textBoxLogin->BorderColor = Color::Red;

MessageWarning->Show("Введите логин");

return;

}

if (textBoxPassword->Text == "") {

textBoxPassword->BorderColor = Color::Red;

MessageWarning->Show("Введите пароль");

return;

}

if (ComeIn()) {

if (SwitchRemember->Checked == true) {

CreateLogFile();

}

if (greeting) {

animation\_mode = 4;

sec = 4;

}

else {

animation\_mode = 5;

sec = 1;

}

labelWelcome->Text += USER;

SetCenter(pageWelcome, labelWelcome, 1);

timerTransition->Start();

}

}

Boolean auth::ComeIn() {

// Получаем логин и пароль пользователя

String^ cliLogin = textBoxLogin->Text;

String^ cliPassword = textBoxPassword->Text;

// Преобразуем их в std::string

std::string Login = msclr::interop::marshal\_as<std::string>(cliLogin);

std::string Password = msclr::interop::marshal\_as<std::string>(cliPassword);

// Создаем путь к файлу Password - data.bin

String^ folderPath = cliLogin;

String^ filePath = Path::Combine(folderPath, "data.bin");

// Проверяем существует ли пользователь в бд

if (!Directory::Exists(folderPath)) {

MessageError->Caption = "Ошибка";

MessageError->Text = "Пользователя с указанным логином не существует";

MessageError->Show();

return false;

}

// Хэшируем введенный пароль

std::hash<std::string> hasher;

size\_t hashPasswordInput = hasher(Password);

// Открываем файл для чтения в бинарном режиме

FileStream^ fileStream = gcnew FileStream(filePath, FileMode::Open, FileAccess::Read);

BinaryReader^ binaryReader = gcnew BinaryReader(fileStream);

// Читаем данные из файла

size\_t hashPasswordFile = binaryReader->ReadUInt64();

// Закрываем бинарный читатель и файловый поток

binaryReader->Close();

fileStream->Close();

// Сравниваем хэши паролей

if (hashPasswordInput == hashPasswordFile) {

USER = cliLogin;

return true; // Пароль совпадает

}

else {

MessageError->Caption = "Предупреждение";

MessageError->Text = "Введен неверный пароль";

MessageError->Show();

return false; // Пароль не совпадает

}

}

Текст остальных программных модулей представлен в приложении А.

## Входные и выходные данные

Входные данные:

- Тема задачи: Строка, представляющая тему задачи.

- Исходный код на Python: Строка, содержащая код задачи на языке Python.

Выходные данные:

- Название задачи: Строка, представляющая название задачи.

- Решение задачи: Строка, содержащая корректный код на Python для решения задачи.

- Пояснение решения: Строка, содержащая объяснение решения задачи.

- Тесты для задачи: Массив строк, каждая из которых представляет собой тестовый случай для проверки решения.

- Файл решения: Отчёт, содержащий решение задачи и пояснение.

Функциональное тестирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №№ | Модуль / Функция | Шаги выполнения | Ожидаемый результат | Фактический результат |
| 11 | Выполнение задачи | 1. Определить сложность задачи. 2. Выбрать задачу на странице из предложенных, и нажать на нее 3. Написать код в функцию и нажать кнопку проверить | Выполнение задачи, сохранения прогресса. | Результат преставлен на рисунке В.1 и В.2 |
| 22 | Использование книги, пометок | 1 Перейти на страницу с книгой.  2 Ознакомиться с материалом  3 Сделать желаемую пометку из предоставленных. | Ознакомление с материалом, создание пометок и их сохранение. | Результат представлен на рисунке В.3, В.4 и В.5 |
| 33 | Создание отчёта прогресса и продуктивности. | 1 Перейти на окно с статистикой.  2 Нажать на кнопку: «Создать отчёт». | Создание отчёта цены и графика изменения цен | Результат представлен на рисунке В.6, В.7 и В.8. |
| 44 | Ознакомление со справочной системой. | 1 Нажать на кнопку F1 | Открытие и ознакомление со справочной системой | Результат представлен на рисунке В.9 и В.10 |

## Описание справочной системы

Средством разработки для создания справочной системы является Dr.Explain, который позволяет легко создавать справочные системы для программного обеспечения, веб-приложений и других продуктов. С его помощью можно быстро и эффективно документировать функциональность и интерфейс приложений, что делает его очень полезным инструментом для разработчиков и технических писателей.

Справочная система имеет начальную страницу, оглавление раздел 1 и раздел 2.

В 1 разделе описывается меню программного средства, где описывается способ открытия и его назначение.

Во 2 разделе описывается главное окно, где отображается главная информация и реализованы основные функции программного средства.

Справочная система вызывается с помощью специальной кнопки вызова F1 на клавиатуре.

# Применение

## Назначение программного средства

Программное средство «PythonWave» предназначено для изучения языка программирования Python. Пользователи могут использовать это программное средство для обучения, создание отчётов своей продуктивности и получение базы Python. Так же познакомившись с этим языком, пользователь может понять симпатизирует ли ему этот язык в целом.

Ограничения, накладываемые на представленное программное средство, включают следующее: необходимость наличия операционной системы Windows.

## Условия применения

Для функционирования программы требуется операционная система Windows, а также среда разработки Visual Studio для компиляции и запуска программы. Кроме того, необходимо установить Windows Forms, который является частью платформы .NET Framework. Для написания программы на C++ и использования CLI .NET также потребуется соответствующий компилятор и библиотеки.

Процесс инсталляции программного средства включает установку операционной системы Windows, последующую установку Visual Studio с выбором необходимых компонентов, включая поддержку C++ и .NET Framework.

# Заключение

Целью курсового проектирования было создание программного средства с целью обучения пользователя. Оно позволяет выполнять практические задачи, изучать теоретический материал, создавать отчеты, настраивать приложение.

Была поставлена задача реализовать данное программное средство на среде разработке Visual Studio, языке программирования C++ и фреймворка CLI .NET.

В процессе разработки возникли трудностей не возникало.

Достоинством является простое взаимодействие пользователя с программным средством. Удобный и лаконичный интерфейс с анимациями делает работу с программой максимально эффективной и приятной. Пользователю не нужно тратить много времени на изучение сложных инструкций или освоение новых навыков — все действия выполняются легко и быстро.

Недостатком является что приложение является локальным. Пользователь не сможет войти в свой аккаунт на других устройствах, где нет сохраненных файлов.

Улучшение данного программного средства возможно полным переходом на язык программирования C#, так как на нем корректно работают сторонние фреймворки, подключение SQL для сохранения данных пользователя онлайн, доработка задач, их пополнение.

# Список информационных источников

1. Блок-схема [Электронный ресурс]. – Wikipedia, 2024. – Режим доступа: <https://ru.wikipedia.org/wiki/Блок-схема>. – Дата доступа: 14.05.2024.
2. Диаграмма классов [Электронный ресурс]. – Wikipedia, 2024. – Режим доступа: <https://ru.wikipedia.org/wiki/Диаграмма_классов>. – Дата доступа: 16.04.2024.
3. Visual Studio 2019 [Электронный ресурс]. – Microsoft, 2020. – Режим доступа: <https://visualstudio.microsoft.com/ru/vs>. – Дата доступа: 22.04.2020.
4. Обзор NET CLI [электронный ресурс]. – Microsoft, 2024. – Режим доступа: <https://learn.microsoft.com/en-us/dotnet/core/tools/>. – Дата доступа: 21.06.2024.
5. Объектно-ориентированное программирование в С++/ [Роберт Лафоре](https://oz.by/people/more901438.html). – 4-е изд. – СПб. : Питер, 2019. – 928 с.
6. Помощь и обучение по Windows [Электронный ресурс]. – Microsoft,2024. – Режим доступа: <https://support.microsoft.com/ru-ru/windows#WindowsVersion=Windows_10&ID0EBBD=Windows_10>. – Дата доступа: 22.05.2024.
7. Шаляпин Ю.В. Методические указания по выполнению дипломного проекта для учащихся по специальности 2-40 01 01 «Программное обеспечение технологий» / Ю.В. Шаляпин. – Минск : КБП, 2023. – 30 c.