Учреждение образования

«Гродненский государственный политехнический колледж»

Специальность: 2 – 40 01 01 «Программное обеспечение информационных

технологий»

Специализация: 2 – 40 01 01 35 «Программное обеспечение обработки

экономической и деловой информации»

Предмет: «Технология разработки программного обеспечения»

Группа: ПЗТ – 40

**ТЕХНО-РАБОЧИЙ ПРОЕКТ**

Техническое задание на разработку приложения для чтения текстовых файлов (книг) “ReadPeak”

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

Разработал А.А.Стельмашук  
Руководитель проекта О.И. Кизер

2024

**Содержание**

Введение 4

[1 Анализ](#_Toc406878074) задачи 6

[1.1 Постановка задачи](#_Элементы_синтаксиса) 6

1.2 Диаграмма вариантов использования 8

1.3 Выбор модели жизненного цикла ПО 8

[1.4](#_Toc406878086) Инструменты разработки 11

2 Проектирование задачи 13

2.1 Разработка системы меню 13

2.2 Разработка UML-диаграмм 13

2.2.1 Диаграмма последовательности 13

2.2.2 Диаграмма деятельности 13

2.2.3 Функциональная модель 14

2.2.4 Модель данных 14

2.2.5 Диаграмма классов 14

2.2.6 Диаграмма объектов 15

2.3 Разработка пользовательского интерфейса 15

3 Реализация 16

3.1 Руководство программиста 16

3.1.1 Организация данных 16

3.1.2 Структура программы 16

3.1.3 Спецификация программы 17

4 Реализация 18

4.1Тесты на использование 18

4.2 Отчёт о результатах тестирования 18

5 Руководство пользователя 19

5.1 Руководство системного программиста 19

5.2 Запуск программы 19

5.3 Инструкция по работе с программой 19

5.4 Завершение работы с программой 22

Заключение 23

Список использованных источников 24

Приложение А «ДВИ» 25

Приложение Б «Структура главного меню» 27

Приложение В «Диаграмма последовательности» 29

Приложение Г «Диаграмма деятельности» 31

Приложение Д «Функциональная модель» 33

Приложение Ж «Модель данных» 35

Приложение И «Диаграмма классов» 37

Приложение К «Диаграмма объектов» 39

Приложение Л «UX и UI прототипирование» 41

Приложение М «Тест кейсы» 45

Приложение Н «Листинг программы» 47

**Введение**

Целью данного техно рабочего проекта по предмету ТРПО является разработка и проектирование программного продукта – приложения для чтения книг «ReadPeak». Данный программный продукт позволит пользователям наслаждаться чтением книг с высоким качеством отображения текста и широкими возможностями настройки.Далее приведено краткое описание разделов пояснительной записки.

Первый раздел носит название «Анализ задачи». В нем можно ознакомиться с постановкой задачи, которая включает в себя: исследование предметной области поставленной задачи, функциональные и нефункциональные требования к программному продукту. Все входные, выходные и условно-постоянные данные тоже будут описаны в первом разделе. В подразделе «Инструменты разработки» рассмотрена среда, в которой создается данный программный продукт. Здесь также установлены минимальные требования к аппаратным характеристикам, обеспечивающим правильное функционирование поставленной задачей. В анализе задачи описана диаграмма вариантов использования, процесс выбора модели жизненного цикла программного обеспечения.

В разделе «Проектирование задачи» рассмотрены основные аспекты разработки программного продукта. Здесь можно узнать об организации данных в контексте среды разработки. В данном разделе четко описан пользовательский интерфейс с помощью UX дизайна.

«Реализация» – это третий раздел пояснительной записки, в котором описываются элементы и объекты, которые использованы при реализации данного приложения. В этом разделе четко описаны функции пользователя и их структура.

Четвертый раздел – «Тестирование». В нем описано функциональное тестирование данной программы, смоделированы все основные действия пользователя при работе с программой.

В разделе «Руководство пользователя» описано назначение, область применения, среда функционирования разработанной программы.

«Заключение» содержит краткую формулировку задачи, результаты проделанной работы, описание использованных методов и средств, описание степени автоматизации процессов на различных этапах разработки.

В разделе «Список использованных источников» приведен список используемых при разработке источников.

В приложении А предоставлена диаграмма вариантов использования.

В приложении Б предоставлена структура главного меню приложения.

В приложении В предоставлена диаграмма последовательности.

В приложении Г предоставлена диаграмма деятельности.

В приложении Д предоставлена функциональная модель.

В приложении Ж предоставлена модель данных.

В приложении И предоставлена диаграмма классов.

В приложении К предоставлена диаграмма объектов.

В приложении Л предоставлены UX и UI прототипирование.

В приложении М предоставлены тест-кейсы.

**1 Анализ задачи**

* 1. **Постановка задачи**

Наименование задачи: разработка приложения для чтения текстовых файлов (книг), “ReadPeak”.

Цель разработки: создание программного продукта, который позволяет пользователям удобно и эффективно читать текстовые файлы с книгами на компьютере

Назначение: приложение предназначено для использования читателями, которые хотят иметь доступ к электронным версиям книг и удобно читать их на персональных устройствах.

Периодичность использования: по мере необходимости

Источники и способы получения данных:с внешних накопителей, с помощью сети Интернет

Обзор существующих аналогичных ПП:Существует множество читалок для текстовых файлов, включая известные программные продукты, такие как Adobe Acrobat Reader, Amazon Kindle, Apple iBooks и другие. Каждая из них имеет свои особенности и возможности для чтения и управления электронными книгами.

Описание перечня функций и задач, которые должен выполнять будущий ПП:

Пользователь:

1. Воспроизведение файлов различных форматов, включая FB2, EPUB, PDF, DOC. Их поиск в устройстве.
2. Поиск книг по названию, автору, жанру и другим параметрам.
3. Добавление книг в каталоги (избранное либо созданные самостоятельно) и их удаление.
4. Оставление заметок, оценок, закладок и выделений.
5. Редактирование и сохранение содержимого книги.
6. Быстрый просмотр содержания по томам, главам, разделам.
7. Просмотр информации о книгах, включая описание и обложку.
8. Возможность настройки внешнего вида интерфейса (цветовая схема, настройки подсветки, шрифт и т.п.).
9. Создание личного кабинета и синхронизация его с остальными устройствами.

Входной информацией будет являться : книги различных форматов, сохраненные на компьютере и другом устройстве пользователя; информация о книгах (название, автор, жанр, обложка, описание и т.д.). Формы представления: файлы с данными (например, в формате JSON или XML); информация о пользователе (книги, каталоги, избранное). Формы представления: файлы с данными (например, в формате JSON или XML).

Выходной информацией будет являться: воспроизведение текстовых файлов с возможностью гибкой настройки под свои удобства; отображение личного кабинета, заметок, закладок, избранного, оценок; отображение информации о текущей книге, включая ее содержимое, ее содержание, автора, обложку и другие метаданные; результаты поиска, сортировки книг по заданным параметрам.

Условно-постоянной информацией будет являться: настройки читалки, такие как язык интерфейса, цветовая схема, размеры; история, включающая информацию о последних читанных книгах, и т.п.; Жанры книг, авторы книг

К разрабатываемому приложению предъявляются следующие требования:

Требования к применению:

Пользовательский интерфейс должен быть интуитивным, удобным в использовании; документация должна быть четкой, информативной и понятной, содержать полную инструкцию по использованию приложения и решению возможных проблем, а также предоставлять списка горячих клавиш и советы по использованию.

Требования к производительности:

Время отклика на действия пользователя (например, поиск книг) должно быть минимальным; приложение должно эффективно использовать ресурсы (память, процессорное время) для обеспечения быстрой работы.

Требования к реализации:

ПП должно быть разработано с помощью таких средств и языков программирования, позволяющие разрабатывать кроссплатформенные приложения не меняя язык.

Требования к надёжности:

Приложение должно иметь высокую степень надежности, обеспечивать минимальное влияние сбоев на работу системы; в случае сбоев, приложение должно иметь возможность восстановления и продолжения работы без потери данных или повреждения файлов.

Требования к интерфейсу: На главной странице должны быть расположены недавно читавшиеся книги, должны быть расположены кнопки личного кабинета, выборки книги с устройства, открытия списка имеющихся книг, кнопка настройки приложения.

**1.2 Диаграмма вариантов использования**

Диаграмма вариантов использования – диаграмма, отражающая отношения между актёрами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне.

В разрабатываемом проекте можно выделить одного актёра: пользователь.

Для пользователя доступны следующие варианты использования: воспроизведение файлов различных форматов, редактирование сборников, поиск/фильтрация книг, манипуляция с книгами, манипуляции со сборниками, настройки приложения, настройки личного кабинета.

Разработанная диаграмма вариантов использования представлена в приложении А.

**1.3 Выбор модели жизненного цикла ПО**

С помощью нескольких таблиц и ответов на вопросы возможно выяснить модель жизненного цикла программного обеспечения.

Выбор модели жизненного цикла на основе характеристик требований находится в таблице 1.

Таблица 1 **–** Выбор модели жизненного цикла на основе характеристик требований

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории требований | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли требования к проекту легко определимыми и реализуемыми? | Да | Да | Да | Нет | Нет | Нет |
| 2. | Могут ли требования быть сформулированы в начале ЖЦ? | Да | Да | Да | Да | Нет | Нет |
| 3. | Часто ли будут изменяться требования на протяжении ЖЦ? | Нет | Нет | Нет | Нет | Да | Да |
| 4. | Нужно ли демонстрировать требования с целью их определения? | Нет | Нет | Да | Нет | Да | Да |
| 5. | Требуется ли проверка концепции программного средства или системы? | Нет | Нет | Да | Нет | Да | Да |
| 6. | Будут ли требования изменяться или уточняться с ростом сложности системы (программного средства) в ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 7. | Нужно ли реализовать основные требования на ранних этапах разработки? | Нет | Нет | Да | Да | Да | Да |

Вычисления: 4 за каскадную, 4 за V- образную, 5 за RAD, 5 за инкрементную, 3 за быстрого прототипирования и 3 за эволюционную.

Таблица 2 **–** Выбор модели жизненного цикла на основе характеристик команды разработчиков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории команды разработчиков  проекта | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли проблемы предметной области проекта новыми для большинства разработчиков? | Нет | Нет | Нет | Нет | Да | Да |
| 2. | Являются ли инструментальные средства, используемые в проекте, новыми для большинства разработчиков? | Да | Да | Нет | Нет | Нет | Да |
| 3. | Изменяются ли роли участников проекта на протяжении ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 4. | Является ли структура процесса разработки более значимой для разработчиков, чем гибкость? | Да | Да | Нет | Да | Нет | Нет |
| 5. | Важна ли легкость распределения человеческих ресурсов проекта? | Да | Да | Да | Да | Нет | Нет |
| 6. | Приемлет ли команда разработчиков оценки, проверки, стадии разработки? | Да | Да | Нет | Да | Да | Да |

Вычисления: 3 за каскадную, 3 за V-образную, 4 за RAD, 2 за инкрементную, 1 за быстрого прототипирования и 1 за эволюционную.

Таблица 3 **–** Выбор модели жизненного цикла на основе характеристик коллектива пользователей

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории коллектива пользователей | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Будет ли присутствие пользователей ограничено в ЖЦ разработки? | Да | Да | Нет | Да | Нет | Да |
| 2. | Будут ли пользователи оценивать текущее состояние программного продукта (системы) в процессе разработки? | Нет | Нет | Нет | Да | Да | Да |
| 3. | Будут ли пользователи вовлечены во все фазы ЖЦ разработки? | Нет | Нет | Да | Нет | Да | Нет |
| 4. | Будет ли заказчик отслеживать ход выполнения проекта? | Нет | Нет | Нет | Нет | Да | Да |

Вычисления: 3 за каскадную, 3 за V-образную, 3 за RAD, 2 за инкрементную, 1 за быстрого прототипирования и 1 за эволюционную.

Таблица 4 **–** Выбор модели жизненного цикла на основе характеристик типа проектов и рисков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории типов проекта и рисков | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Разрабатывается ли в проекте продукт нового для организации направления? | Нет | Нет | Нет | Да | Да | Да |
| 2. | Будет ли проект являться расширением существующей системы? | Да | Да | Да | Да | Нет | Нет |
| 3. | Будет ли проект крупно- или среднемасштабным? | Нет | Нет | Нет | Да | Да | Да |
| 4. | Ожидается ли длительная эксплуатация продукта? | Да | Да | Нет | Да | Нет | Да |
| 5. | Необходим ли высокий уровень надежности продукта проекта? | Нет | Да | Нет | Да | Нет | Да |
| 6. | Предполагается ли эволюция продукта проекта в течение ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 7. | Велика ли вероятность изменения системы (продукта) на этапе сопровождения? | Нет | Нет | Нет | Да | Да | Да |
| 8. | Является ли график сжатым? | Нет | Нет | Да | Да | Да | Да |
| 9. | Предполагается ли повторное использование компонентов? | Нет | Нет | Да | Да | Да | Да |
| 10. | Являются ли достаточными ресурсы (время, деньги, инструменты, персонал)? | Нет | Нет | Нет | Нет | Да | Да |

Вычисления: 6 за каскадную, 5 за V-образную, 7 за RAD, 4 за инкрементную, 5 за быстрого прототипирования и 5 за эволюционную.

Общий итог: 16 за каскадную, 15 за V-образную, 19 за RAD, 13 за инкрементную, 10 за быстрого прототипирования, 10 за эволюционную.  
 В итоге заполнения табл. 1 – 4 наиболее подходящей является RAD модель

**1.4 Инструменты разработки**

Для разработки проекта был выбран язык программирования C# и среда разработки Visual Studio, так как C# является мощным объектно-ориентированным языком с широким набором библиотек. Visual Studio предоставляет широкий набор инструментов для быстрой разработки приложений и удобного визуального проектирования.

Преимущества от проектирования в среде Windows с помощью C#:

1 Обеспечение согласованности проекта и его реализации благодаря мощным инструментам отладки и тестирования.

2 Увеличение производительности разработки за счет использования готовых библиотек и компонентов, а также возможности интеграции с другими сервисами и API.

C# — это мощный и современный язык программирования, который под-держивает объектно-ориентированное программирование, что позволяет создавать чистый и поддерживаемый код. Использование одного языка для разработки как логики приложения, так и пользовательского интерфейса упрощает процесс разра-ботки.

C# и .NET имеют обширную экосистему библиотек и инструментов, что позволяет быстро находить готовые решения для распространенных задач, следо-вательно это может значительно ускорить процесс разработки и улучшить каче-ство приложения.

Также C# и .NET имеют большое и активное сообщество разработчиков, таким образом можно легко найти помощь, примеры кода и ресурсы для обуче-ния, что особенно полезно для решения проблем и получения новых идей

Также для разработки программы необходимы:

* Google-браузер – необходим для нахождения информации и фоновых изображений;
* Word 2024 – необходим для написания пояснительной записки;
* Power Point 2024 – необходим для создания отчётной презентации;
* Draw io – необходим для создания диаграмм.

При разработке данного программного продукта был использован

компьютер со следующими характеристиками:

* процессор Intel(R) Core(TM) i5-8350U CPU @ 1.70GHz;
* ОЗУ: 8Gb;
* память: SDD 512Gb;
* ОС – Windows 10.

**2 Проектирование задачи**

**2.1 Разработка системы меню**

Эффективная структура главного меню и продуманная система навигации являются основой удобства использования приложения для чтения книг и влияют на впечатление пользователей о проекте. Грамотная организация меню и навигационных элементов помогает пользователям быстро находить нужную информацию, легко перемещаться между разделами и книгами, а также улучшает восприятие контента.

На этапе разработки структуры главного меню важно учитывать потребности целевой аудитории, принципы юзабилити, а также технические особенности проекта, чтобы создать логичную и интуитивно понятную навигацию. В этом разделе рассмотрены этапы проектирования структуры приложения, основные подходы к организации меню и варианты навигационных решений, направленных на улучшение взаимодействия пользователей с приложением для чтения книг.

Полную структуру приложения можно рассмотреть в приложении Б.

**2.2 Разработка UML-диаграмм**

**2.2.1 Диаграмма последовательности**

В первом разделе «1.1 Постановка задачи» были описаны функциональные требования к проекту. На их основе построена диаграмма последовательности. Диаграмма последовательности – это диаграмма, предназначенная для моделирования взаимодействия объектов системы во времени, а также обмена сообщениями между ними. Графические диаграммы последовательности имеют два представления. Одно – слева направо – в виде вертикальных линий, которые соответствуют линии жизни отдельного участка взаимодействия. Второе –вертикальная временная ось, направленная сверху вниз. На диаграмме отображен процесс регистрации.

Диаграмму последовательности можно рассмотреть в приложении В.

**2.2.2 Диаграмма деятельности**

Диаграмма деятельности – это диаграмма, предназначенная для моделирования динамических аспектов системы, описывая последовательность выполняемых действий и поток управления между ними. Она помогает визуализировать логику выполнения процесса или сценария, показывая, как действия связаны друг с другом и каким образом происходит переход от одного действия к другому. На диаграмме отображен процесс оформления заказа на аренду техники, включая шаги и возможные развилки в процессе.

Диаграмму деятельности рассмотрим в приложении Г.

**2.2.3 Функциональная модель**

Функциональная модель представляет собой описание функциональности системы. Она включает в себя множество диаграмм и моделей, таких как диаграммы потоков данных (DFD), которые показывают, как информация движется через систему и как она обрабатывается. Основная цель функциональной модели - показать, как система выполняет свои функции и взаимодействует с внешними компонентами и пользователями. Это помогает идентифицировать и устранить потенциальные проблемы на ранних стадиях разработки.

Функциональную модель рассмотрим в приложении Д.

**2.2.4 Модель данных**

Модель данных описывает, как данные организованы и хранятся в системе. Она включает схемы баз данных, связи между таблицами, а также правила, управляющие данными. Модель данных важна для обеспечения целостности и консистентности данных в приложении. Она помогает разработчикам понять, как данные должны быть структурированы, и обеспечивает основу для взаимодействия с базами данных. Модель данных также играет ключевую роль в оптимизации производительности и масштабируемости системы.

Модель данных рассмотрим в приложении Ж.

**2.2.5 Диаграмма классов**

Диаграмма классов — это диаграмма, используемая в объектно-ориентированном программировании для визуализации структуры системы. Она показывает классы, их атрибуты, методы и взаимоотношения между ними. Диаграммы классов позволяют разработчикам увидеть, как компоненты системы взаимосвязаны, что помогает в планировании и реализации программного обеспечения. Это важный инструмент для понимания и документирования архитектуры приложения, а также для координации командной работы.

Диаграмму классов рассмотрим в приложении И.

**2.2.6 Диаграмма объектов**

Диаграмма объектов – это диаграмма, предназначенная для моделирования статических связей между объектами системы в конкретный момент времени. В отличие от диаграммы классов, которая отображает классы и их связи в общем виде, диаграмма объектов фокусируется на конкретных экземплярах классов и их связях, показывая состояние системы в определенной ситуации. Связи между объектами показывают, как взаимодействуют конкретные экземпляры классов в ходе выполнения сценария. Диаграмму объектов рассмотрим в приложении К.

**2.3 Разработка пользовательского интерфейса**

Важным элементом проектирования данного программного продукта является описание внешнего интерфейса разрабатываемой игры.

Для разработки визуального дизайна использовались сдержанные, мягкие цвета для удобства использования программного продукта. В ходе разработки были спроектированы дизайны всех страниц игры, как видимых пользователю, так и невидимых. Ранее разработанная структура главного меню расположена на рисунке 1.

Для организации эффективной работы пользователя нужно создать целостный программный продукт данной предметной области, в котором все компоненты будут сгруппированы по функциональному назначению. Все исходные данные будут разделены на несколько групп.

Прототип – это наглядная модель пользовательского интерфейса. В сущности, это «черновик», созданный на основе представления разработчика о потребностях пользователя. Итоговое отображение программы может отличаться от прототипа.

Прототип UX представлен в приложении Л, ссылка на UX: https://www.figma.com/design/YrAzHP4QOZ59FGAMBVeGe6/ux17?node-id=0-1&p=f&t=JlAOx5I1K4Eyl4JM-0

Прототип UI представлен в приложении Л, ссылка на UI: https://www.figma.com/file/qBatGp2qzYA3nRpAIZV4Fk?node-id=0-1&node-type=canvas&t=rq0ooixoCpV4QdsD-0&type=design&mode=design&fuid=1425031726326622792

**3 Реализация**

**3.1 Руководство программиста**

**3.1.1 Организация данных**

Основным средством хранения информации в приложении являются: xml-файлы - это файл общего характера, который сохраняет определенные данные. В данном приложении используется для хранения состояний прочтения книг.

Приложение имеет возможность воспроизводить файлы форматов: EPUB, FB2, PDF, TXT.

* + 1. **Структура программы**

Приложение написано с использованием объектно-ориентированной архитектуры (ООП). Оно состоит из множества классов, каждый из которых выполняет свою уникальную функцию. Классы можно функционально разделить на некоторые группы:

ManageLibrary – классы этой группы (AddBookCommand.cs, ChangeLibraryPathCommand.cs, DeleteBookCommand.cs, EmptyLibraryCommand.cs, MarkFavoriteCommand.cs, RemoveFavoriteMarkCommand.cs, SortCommand.cs) отвечают за манипуляции с библиотеками книг: их добавление\удаление, занесение в избраное, сортировку.

NavigationCommands – классы этой группы (NavigateBackCommand.cs, NavigateMenuCommand.cs, NavigateReadBookCommand.cs) отвечают за навигацию внутри программы

ReadingSettingsCommands – классы этой группы (DecreaseLineSpacingCommand.cs, IncreaseLineSpacingCommand.cs, DecreaseMarginWidthCommand.cs, IncreaseMarginWidthCommand.cs, CommandBase.cs) отвечают за изменение настроек отображения текста в книге

Utils – классы этой группы (ProgressSerializer.cs, SerializableProgress.cs, Serializer.cs) отвечают за вспомогательные специализированные задачи в программе, как например сериализация и десериализация данных.

View – классы этой группы (AllBooksView.xaml.cs, FavoriteBooksView.xaml.cs, MainMenuView.xaml.cs, MainWindow.xaml.cs, ReadBookView.xaml.cs, SettingsView.xaml.cs) отвечают за логику взаимодействия с интерфейсом программы

В приложении также используется библиотека VersOne.Epub для работы с epub файлами.

Эта структура позволяет обеспечить модульность, гибкость и упрощает дальнейшее развитие и сопровождение программного продукта.

* + 1. **Спецификация программы**

Точное название проекта и его состав приводится в Таблице 6.

Таблица 6 –– Спецификация программы

|  |  |
| --- | --- |
| **Имя файла** | **Назначение** |
| ReadPeak.exe | Исполняемый файл проекта |
| book\_icon.ico | Иконка ярлыка приложения. |
| Castle.Core.dll, HtmlAgilityPack.dll, Microsoft.Xaml.Behaviors.dll, Moq.dll, Nancy.dll, Newtonsoft.Json.dll, NServiceBus.Core.dll, NuGet.Common.dll, Prism.dll, VersOne.Epub.dll | Файлы дополнительных подгружаемых библиотек, от которых зависит корректная работа приложения |
| BookList.xml | файл, хранящий название файлов подключенных книг, состояние их прочтения и добавленности в сборники. |
| ReadPeak.deps.json | Cодержит информацию о зависимостях, необходимых для корректного функционирования приложения. |
| ReadPeak.runtimeconfig.json | Файл, который содержит конфигурацию среды выполнения для приложения. |
| ReadPeak.pdb | Файл, содержащий отладочную информацию |
| Library | Стандартная папка, содержащая добавленные пользователем книги |

**4 Тестирование**

**4.1 Тесты на использование**

При разработке данной программы многие возникающие ошибки и недоработки были исправлены на этапе реализации проекта. После завершения испытания реализации программы было проведено тщательное функциональное тестирование. Функциональное тестирование должно гарантировать работу всех элементов программы в автономном режиме.

Разработанные тест-кейсы и статус их выполнения представлены в приложение К.

Расписание работ над проектом представлено в таблице 7.

Таблица 7 – Расписание работ над проектом

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя** | **Дата** | **Деятельность** | **Продолжительность, ч** |
| Стельмашук Аркадий | 01.12.2024 | Разработка тест-кейсов | 14 |
| Стельмашук Аркадий | 01.12.2024 | Проведение тестирования | 14 |

**4.2 Отчёт о результатах тестирования**

Элементы программы были проверены, и было установлено, что все они работают правильно и выполняют задачи, указанные в процедурах.

Статистика по всем дефектам представлена в таблице 8.

Таблица 8 – Статистика по всем дефектам

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Важность | | | |
| Статус | Количество | Низкая | Средняя | Высокая | Критическая |
| Найдено | 0 | 0 | 0 | 0 | 0 |
| Исправлено | 0 | 0 | 0 | 0 | 0 |
| Проверено | 0 | 0 | 0 | 0 | 0 |
| Открыто заново | 0 | 0 | 0 | 0 | 0 |
| Отклонено | 0 | 0 | 0 | 0 | 0 |

**5 Руководство пользователя**

Приложение «ReadPeak» позволит пользователю наслаждаться чтением, получать эстетическое удовольствие и повысить свою продуктивность.

Создаваемое приложение ориентировано на пользователей всех возрастов.

Приложение не очень требовательно к характеристикам компьютера и может запуститься на очень старых ПК.

Данный программный продукт можно использовать в любое время, так как он не ограничен в использовании.

Загрузка музыкального плеера предполагает наличие свободной памяти в размере 500 МБ.

**5.1 Руководство системного программиста**

Для того, чтобы установить программу нужно запустить установщик setup.exe и следуя инструкциям мастера установки завершить установку программы.

**5.2 Запуск программы**

После установки программы, по двойному нажатию на ярлык «ReadPeak» будет запускаться читалка.

**5.3 Инструкция по работе с программой**

При запуске приложения пользователя встречает главное меню, в котором пользователь может наблюдать добавленные книги, список избранного, кнопки добавления книг и продолжения чтения. Главное меню представлено на рисунке 1.

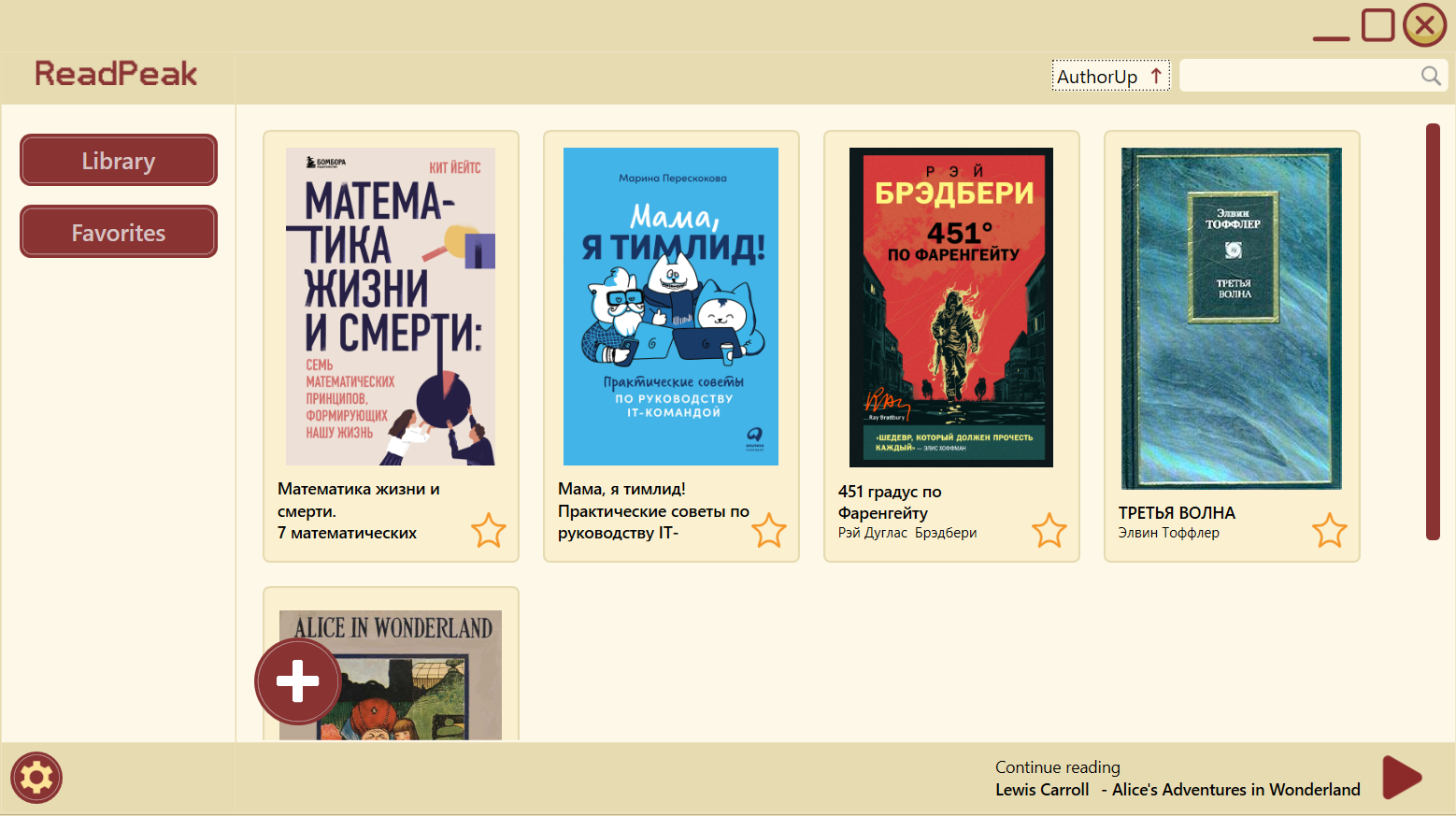


Рисунок 1 – Главное меню.

До начала чтения книг пользователь сначала должен добавить их в приложение, для этого пользователь должен навести курсор на кнопку с плюсиком, затем в появившемся диалоговом окне выбрать нужный ему файл.

Кнопка добавления файла представлена на рисунке 2.



Рисунок 2 – Кнопка добавления книги.

Далее добавленный пользователем файл сразу добавиться в список книг.

Список книг представлен на рисунке 3.

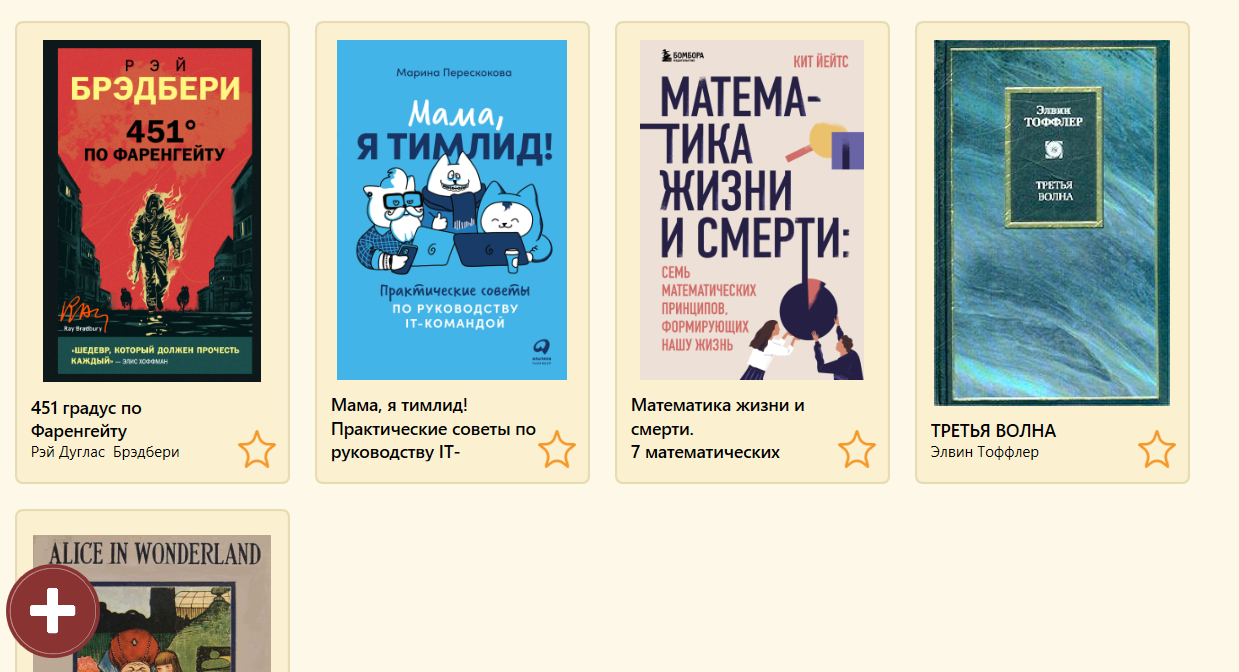


Рисунок 3 – Список треков

Далее для начала чтения книги нужно кликнуть на него левой кнопкой мыши.

Если же у пользователя уже есть начатая книга, то об этом будет показано в правом нижнем углу приложения с предложением продолжить чтение данной книги.

Предложение продолжения чтения представлен на рисунке 4

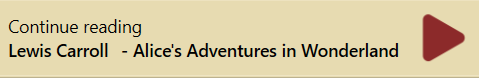


Рисунок 4 – Предложение продолжения чтения

Для организации книг в сборники, пользователь имеет возможность удалять ненужные ему треки из сборника. Для этого следует нажать правой кнопкой по книге и выбрать кнопку «delete». Представлено на рисунке 5.

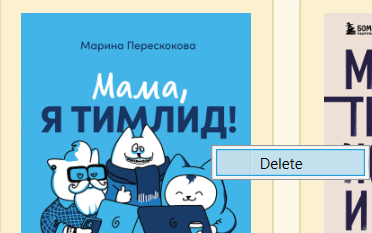


Рисунок 5 – Кнопка удаления книги из сборника

Также, на главном меню имеется выпадающий список для выбора метода сортировки книг (в алфавитном либо обратном порядке, по названию книги либо автора) и компонента поиска книг. Представлено на рисунке 6

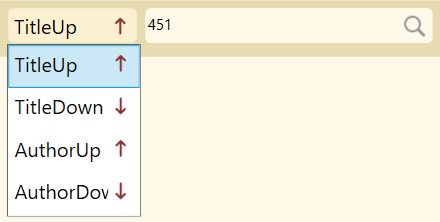


Рисунок 6 – Сортировка и поиск книг

Зайдя в книгу, пользователь может изменить вид отображения страниц (одна страница либо две) и масштаб книги (с помошью ползунка). Представлено на рисунке 7



Рисунок 7 – Вид отображения страниц вместе с масштабом

Также в выпадающем меню настроек внутри книги, пользователь имеет возможность изменить расстояние между строками, расстояние от краев, вид выравнивания, отображаемый шрифт, а также цвет заднего фона книги. Меню представлено на рисунке 8.

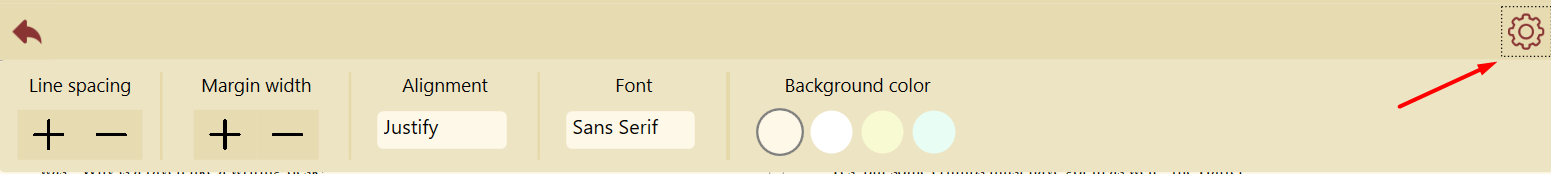


Рисунок 8 – Выпадающее меню настроек отображения

Плюсом к этому, находясь в книге имеется возможность поиска ключевых слов во всем тексте. Представлено на рисунке 9.



Рисунок 9 – Поиск

**5.4 Завершение работы с программой**

Завершить работу проигрывателя можно следующим способом: в главном меню, верхнем правом углу, присутствует кнопка «Выйти», после нажатия на которую приложение закрывается.

**Заключение**

Целью данного техно-рабочего проекта по предмету ТРПО является разработка и проектирование программного продукта – приложения для чтения книг «ReadPeak». Этот программный продукт позволяет пользователям наслаждаться чтением с высоким качеством отображения текста и широкими возможностями настройки. ReadPeak предоставляет удобные инструменты для организации книжной библиотеки и управления коллекциями, улучшая читательский опыт пользователей.

В ходе реализации поставленной задачи были получены знания по использованию среды разработки Visual Studio, а также закреплены знания курса предмета «Технология разработки программного обеспечения». Поставленная задача реализована частично. В соответствии с задуманными требованиями создано и протестировано 15 функций. В ходе тестирования все исключительные ситуации были обработаны. Проект работает без сбоев и ошибок. В поставленной задаче был реализован простой и понятный пользовательский интерфейс.

Исходя из этого, можно сделать вывод, что частичная реализация программы прошла успешно.

**Список использованных источников**

1. Руководство по созданию приложения на WPF на C# в Visual Studio [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/wpf/– Дата доступа: 07.12.2024.

2. XAML overview (WPF .NET) [Электронный ресурс]. – Режим доступа:https://learn.microsoft.com/en/us/dotnet/desktop/wpf/xaml/?view=netdesktop-9.0 – Дата доступа: 12.12.2024.