Частное учреждение образования

«Колледж бизнеса и права»

ДОПУЩЕН К ЗАЩИТЕ

Заместитель директора по учебной работе

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.В.Малафей

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023

РЕАЛИЗАЦИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ УЧЕТА ПРОДАЖ КНИЖНОЙ ПРОДУКЦИИ ООО «АРБИС-БЕЛ»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

ДП Т.992136.401

Председатель цикловой комиссии

Руководитель проекта

Консультант по экономической части

Консультант по охране труда

Обучающийся

Рецензент

( Т.Г.Багласова )

( Л.В.Басалыга )

( Е.А.Андреева )

( В.С.Кудласевич)

( Е.А.Чужавко )

( )

2023

СОДЕРЖАНИЕ

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3 инженер ОТ

*ДП Т.992136.401*

*ОП Т.992022*

*ОП Т.992022*

Разраб.

Чужавко Е.А.

Провер.

Т.контр.

Н.контр.

Утверд.

РЕАЛИЗАЦИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ УЧЕТА ПРОДАЖ КНИЖНОЙ ПРОДУКЦИИ ООО «АРБИС-БЕЛ»

Лит.

Листов

КБиП

у

Введение 4

1 Описание задачи 6

1.1 Анализ предметной области 6

1.2 Постановка задачи 6

2 Проектирование мобильного приложения 7

2.1 Проектирование модели 7

2.2 Требования к мобильному приложению 8

2.3 Структура мобильного приложения 8

2.4 Проектирование макета мобильного приложения 9

2.5 Защита и сохранность данных 22

2.6 Организация и ведение информационной базы (модели) 23

3 Реализация мобильного приложения 28

3.1 Программно-технические средства, необходимые для разработки приложения 28

3.2 Описание разделов приложения 29

3.3 Описание используемых функций и процедур 29

3.4 Функциональное тестирование 30

4 Применение 34

4.1 Общие сведения 34

4.2 Назначение программного средства 34

5 Охрана труда и окружающей среды 36

5.1 Правовые, нормативные, социально-экономические и организационные вопросы охраны труда. 36

5.2 Анализ метеорологических условий на рабочем месте 39

5.3 Пожарная безопасность 41

5.4 Охрана окружающей среды 43

6 Экономический раздел 45

6.1 Технико-экономическое обоснование разработки программного средства 45

6.2 Составление плана по разработке программного продукта 45

6.3 Расчет затрат на разработку программного продукта 47

6.4 Расчет экономического эффекта у разработчика и пользователя программного продукта 52

6.4.1 Экономический эффект у разработчика программного продукта 52

6.4.2 Экономический эффект у пользователя программного продукта 53

6.5 Расчет экономического эффекта для свободной реализации на рынке ИТ 56

Заключение 57

Список информационных источников 58

ПРИЛОЖЕНИЕ А Текст программных модулей 59

ПРИЛОЖЕНИЕ Б Результат работы приложения 77

Басалыга Л.В.

Багласова Т.Г.

Басалыга Л.В.

Багласова Т.Г.

77

Введение

Целью дипломного проекта на тему «Реализация мобильного приложения для автоматизации учета продаж книжной продукции ООО «Арбис-Бел» является разработка мобильного приложения для возможности оформления заказов книжной продукции, также для ведения учёта об оформленных заказах, просмотра статистики и управления заказами. Мобильное приложение предполагает функционал вокруг двух групп пользователей:

* первая группа – покупатель. Для покупателя предполагается просмотр каталога товаров, подробный просмотр информации о товарах, просмотр и возможность написать отзывы, оформление заказов, личный кабинет, который включает в себя просмотр информации о пользователе и об оформленных заказах;
* вторая группа – администратор. Для администратора предполагается просмотр статистики, добавление книг, контроль за отзывами, просмотр и контроль за заказами.

Книги являются источниками информации. Эта информация может нести как развлекательный характер, так и научный характер. Информация, которая находится в виде книги имеет ряд преимуществ таких как:

* физический объект – книга доступна всегда и нет необходимости искать информацию в интернете;
* узкая область информации – книги написаны на определённую тематику, что означает в отсутствии необходимости искать другие источники информации.

Актуальность задачи в том, что приложение может повлиять на рынок книг. Это влияние может создать развитие конкуренции на рынке, что приведёт к увеличению количества потенциальных покупателей, увеличению потребности книжной продукции. Приложение даст удобство к доступу информации, что происходит благодаря использованию мобильных устройств.

Пояснительная записка к дипломному проекту состоит из шести разделов и графической части.

В первом разделе «Описание задачи» раскрывается организационная сущность задачи, описывается предметная область и круг задач, которые должны быть автоматизированы. Описывается задача, перечисляются основные функции программы. Так же указываются существующие аналоги.

Во втором разделе «Проектирование мобильного приложения» перечисляются требования к внешнему виду приложения, контенту и мобильному устройству, структуре, макету, защите данных, приводится обоснование выбранного инструмента разработки, описание и взаимосвязь таблиц.

В третьем разделе «Реализация мобильного приложения» производится описание функций мобильного приложения, их взаимосвязи и листинги всех модулей. Так же описываются средства разработки и обоснования выбора технологий. Описываются требования к техническим средствам для проведения испытаний, требования к характеристикам программы применительно к условиям эксплуатации, требования к информационной и программной совместимости. Предоставляются результаты функционального тестирования.

Четвёртый раздел «Применение» предназначен для описания сведений о назначении программного средства и области его применения. Так же описывается потенциальная аудитория, средства защиты и разграничения доступа к данным.

В пятом разделе «Охрана труда и окружающей среды» рассматриваются: правовые, нормативные, социально-экономические и организационные вопросы охраны труда; разработка мер по нормализации и защиты от ультрафиолетового излучения; пожарная безопасность; охрана труда и окружающей среды.

В шестом экономическом разделе производится технико-экономическое обоснование разработки программного средства, составление плана по разработке программного средства, определение цены программного средства, описывается экономическая эффективность разработки.

В заключении описывается выполнение поставленной задачи, степень соответствия проектных решений заданию, причины несоответствия, если таковые имеются.

В приложении А представлен текст программных модулей.

В приложении Б представлены результаты работы приложения.

Графическая часть содержит следующие диаграммы: структура мобильного приложения, диаграмма вариантов использования, диаграмма классов.

1 Описание задачи

* 1. Анализ предметной области

Предметной областью решаемой задачи является книжный магазин.

В современном мире человек зависим от передовых технологий, которые позволяют человеку получать любую необходимую ему информацию. Информационные технологии являются не только инструментом для получения знаний, но также помощником, который позволяет не тратить время для того, чтобы пойти в магазин и купить только товар, который имеется в наличии. Это позволяет магазинам показать весь свой ассортимент потенциальному покупателю. Для того, чтобы не тратить человеку время на поиск книги, люди используют возможности информационных технологий. На помощь приходят интернет-магазины или мобильные приложения. Хоть вся информация есть в интернете, но все равно есть люди, которым необходима книга как физический объект. Книга – источник всяческих знаний. Она заставляет человека мыслить, воспитывать собственное мнение, развивать воображение.

Существует множество аналогов онлайн-магазинов. Наиболее популярным магазином книг является веб-сайт «oz.by». Сайт имеет широкий ассортимент книг и не только. Данный сайт позволяет искать книги по различным критериям, смотреть отзывы на товары и сохранять их в избранное. В данном сайте сделан каталог по самым продаваемым и новым книгам. Данную идею хотелось бы реализовать в создаваемом программном средстве.

Главным плюсом этого сайта является то, что это веб-сайт и им можно пользоваться на всех платформах, а главный минус – несовременный и сложный интерфейс сайта.

1.2 Постановка задачи

После анализа предметной области необходимо решить следующие задачи: необходимо создать лаконичный интерфейс для ввода или вывода данных покупателя, выбора книг для заказа, возможности оформления отзывов к книге, просмотра всех оформленных заказов и информации о них, возможности создания запросов в службу поддержки и просмотра ответа на них, также для сотрудника просмотра статистики, учета книг и заказов, добавления книг, контроля за отзывами, возможности давать ответы на вопросы в службе поддержки, формирования отчетов по пользователям, книгам, заказам за период.

1. Проектирование мобильного приложения
   1. Проектирование модели

На основании проведенного анализа предметной области и выявленных функциональных возможностей, необходимо будет построить модель, которая бы отображала функциональную структуру объектов программного средства, действия, производимые ими и связь между этими действиями, а также определить инструменты разработки, которые будут использоваться в ходе написания программного средства.

В качестве инструмента для построения такой модели программного средства будет выбран унифицированный язык моделирования. Данный язык моделирования предназначен для специфицирования, визуализации, конструирования и документирования программных средств.

В рамках языка UML все представления о модели сложной системы фиксируются в виде специальных графических конструкций – диаграмм. В терминах языка UML определены следующие виды диаграмм: диаграмма вариантов использования, диаграмма классов, диаграмма деятельности.

Перечень этих диаграмм представляет собой неотъемлемую часть графической нотации языка UML, сам процесс объектно-ориентированного программирования (ООП) неразрывно связан с процессом построения этих диаграмм.

Суть диаграммы вариантов использования состоит в следующем: проектируемая система представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью вариантов использования.

Варианты использования описывают как взаимодействие между пользователем и сущностью, так и реакции сущности на получение отдельных сообщений от пользователей и восприятие этих сообщений за пределами сущности. Варианты использования могут включать в себя описание особенностей способов реализации сервиса и различных исключительных ситуаций, таких как корректная обработка ошибок системы. Множество вариантов использования в целом должно определять все возможные стороны ожидаемого поведения системы.

Диаграмма вариантов использования, отражающая варианты использования приложения для пользователя системы представлена в графической части на листе 1.

Диаграмма классов служит для предоставления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений. На данной диаграмме не указывается информация о временных аспектах функционирования системы.

Диаграмма классов представляет собой некоторой граф, вершинами которого являются элементы типа «классификатор», которые связаны различными типами структурных отношений. Когда говорят о данной диаграмме, имеют в виду статическую структурную модель проектируемой системы. Поэтому диаграмму классов принято считать графическим представлением таких структурных взаимосвязей логической модели системы, которые не зависят или инвариантны от времени. Диаграмма классов представлена в графической части на листе 2.

* 1. Требования к мобильному приложению

Основными требованиями к внешнему виду приложения является использование контрастных цветов, современных элементов дизайна.

Основными минимальными требованиями, выдвигаемыми к аппаратному обеспечению мобильного телефона, являются:

* процессор с частотой не менее 1.2 ГГц;
* оперативная память 1 Гбайт и более;
* внутренняя память не менее 8 Гбайт.

Мобильный телефон должен быть с установленной операционной системой (ОС) Android версии не ниже 5.0.0, так как разработка мобильного приложения не подразумевает версии ниже из-за усложнения процесса разработки.

Все страницы приложения должны иметь полосу прокрутки для устройств с маленькой диагональю экрана.

Приложение должно работать только в вертикальном режиме.

* 1. Структура мобильного приложения

Каждое мобильное приложение имеет структуру. Первое, что видит при запуске пользователь – экран загрузки. После него появляется стартовое окно, которое предлагает 2 варианта. Первый вариант – войти в аккаунт. Второй вариант – авторизация.

Если пользователь является покупателем, то после входа ему откроется каталог книжной продукции. Если пользователь решит нажать на любую книгу из каталога, то откроется подробная информация о книге с возможностью просмотреть отзывы к книге. Покупатель также может выбрать корзину в навигационном меню после входа, где он увидит выбранные книги к заказу. На книги также можно нажать, чтобы просмотреть информацию и отзывы. Если покупатель нажмёт на иконку профиля в навигационном меню, то ему откроется окно, где можно будет просмотреть информацию о профиле, заказах, и обращения в службу поддержки.

Если пользователь является администратором, то после входа ему откроется окно с панелью администратора, где можно просмотреть статистику, заказы, книги, отзывы, запросы в поддержку, добавить книгу.

Структура мобильного приложения представлена в графической части на листе 3, которая показывает взаимосвязи между различными компонентами приложения.

* 1. Проектирование макета мобильного приложения

У мобильного приложения должна быть возможность открытия и масштабируемости на любом Android-устройстве, с любым разрешением экрана и соотношением сторон, с версией Android не менее 5.0.0.

На начальной странице будет показываться название приложения и кнопки авторизации и регистрации, которые позволяют выбрать действие в приложении. Прототип начальной страницы представлен на рисунке 2.1.



Рисунок 2.1 – Прототип страницы «Главная»

При нажатии на кнопку «Войти» будет открываться контекстное меню с полями для ввода данных для входа. Прототип меню после кнопки «Войти» представлен на рисунке 2.2.



Рисунок 2.2 – Прототип контекстного меню после кнопки «Войти»

При нажатии на кнопку «Регистрация» будет открываться контекстное меню с полями для ввода данных для входа. Прототип меню после кнопки «Регистрация» показан на рисунке 2.3.



Рисунок 2.3 – Прототип контекстного меню после кнопки «Регистрация»

После успешной авторизации или регистрации открывается страница с каталогом. Прототип страницы с каталогом представлен на рисунке 2.4



Рисунок 2.4 – Прототип страницы с каталогом

При нажатии на книгу открывается страница с информацией о книге. Прототип страницы с информацией о книге представлен на рисунке 2.5.



Рисунок 2.5 – Прототип страницы с описанием книги

При нажатии на кнопку «Посмотреть отзывы» открывается страница с отзывами к определенной книге, где также монжно оставить отзыв. Прототип страницы с отзывами представлен на рисунке 2.6.

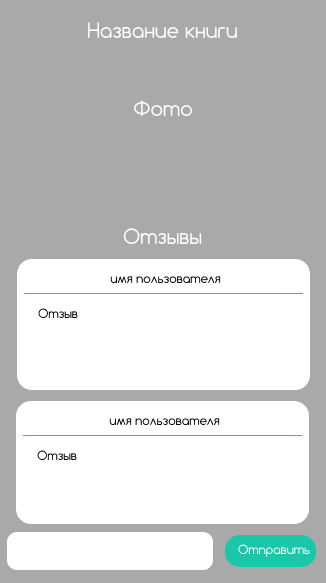


Рисунок 2.6 – Прототип страницы с отзывами

При нажатии на кнопку «Корзина» открывается страница с корзиной товаров. Прототип страницы с корзиной представлен на рисунке 2.7.



Рисунок 2.7 – Прототип страницы с каталогом

При нажатии на определенную книгу откроется информация о книге. Прототип страницы с информацией о книге представлен на рисунке 2.8.

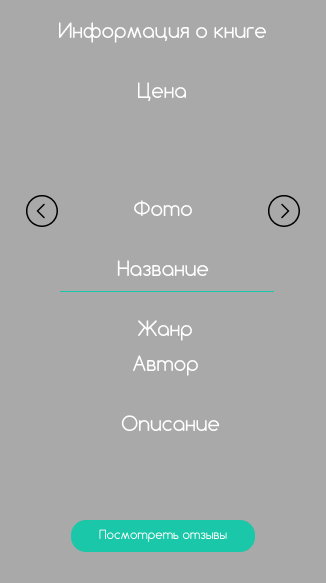


Рисунок 2.8 – Прототип страницы с информацией о книге

При нажатии на кнопку «Профиль» открывается страница с корзиной товаров. Прототип страницы с корзиной представлен на рисунке 2.9.



Рисунок 2.9 – Прототип страницы с профилем

При нажатии на кнопку «Профиль» открывается окно с информацией о пользователе. Прототип страницы с информацией о пользователе представлен на рисунке 2.10.



Рисунок 2.10 – Прототип страницы с информацией о пользователе

При нажатии на кнопку «Все заказы» открывается окно со списком всех заказов пользователя. Прототип страницы со списком заказов представлен на рисунке 2.11.

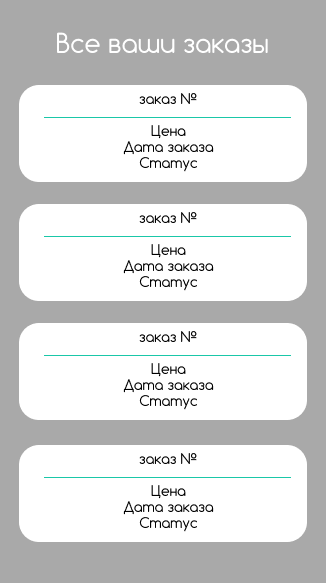


Рисунок 2.11 – Прототип страницы с информацией о пользователе

При нажатии на заказ из списка открывается страница с информацией о заказе. Прототип страницы с информацией о заказе представлен на рисунке 2.12.

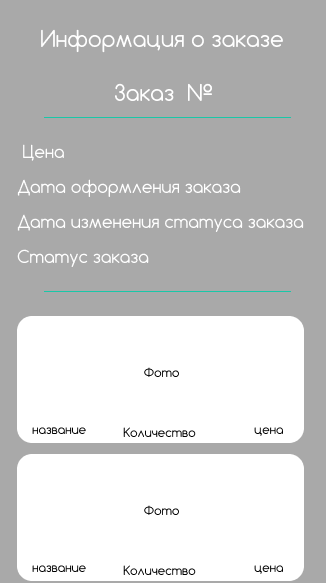


Рисунок 2.12 – Прототип страницы с информацией о пользователе

При нажатии на кнопку «Поддержка» откроется меню поддержки. Прототип страницы с меню поддержки представлен на рисунке 2.13.



Рисунок 2.13 – Прототип страницы с меню поддержки

При нажатии на кнопку «Просмотреть свои запросы» откроется страница, где будут представлены все запросы пользователя в службу поддержки. Прототип страницы со всеми запросами в службу поддержки представлен на рисунке 2.14.

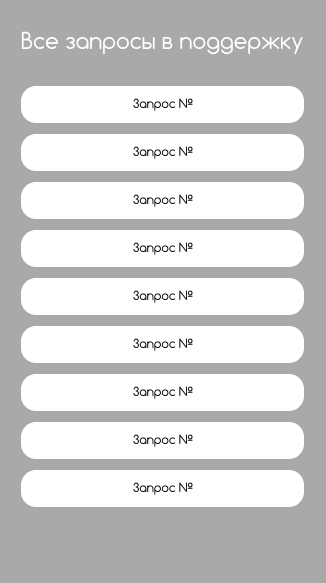


Рисунок 2.14 – Прототип страницы со всеми запросами в службу поддержки

При нажатии на определённый запрос откроется окно, где показан сам вопрос в поддержку и ответ на вопрос, если он имеется. Прототип страницы с вопросом и ответом представлен на рисунке 2.15.



Рисунок 2.15 – Прототип страницы с вопросом и ответом

При нажатии на кнопку «Создать запрос в поддержку» откроется страница с полем для ввода вопроса. Прототип страницы с вводом поля для вопроса в службу поддержки показан на рисунке 2.16.



Рисунок 2.16 – Прототип страницы с вводом поля для вопроса в службу поддержки

При входе в приложение в лице администратора открывается меню администратора. Прототип страницы с меню администратора представлен на рисунке 2.17.

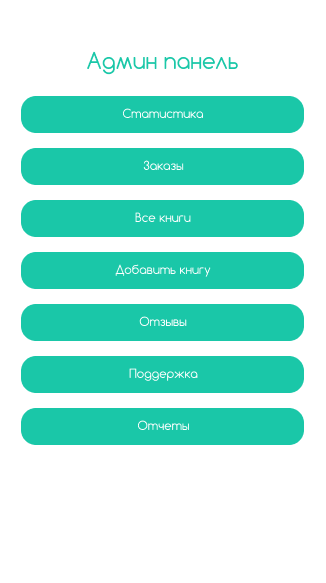


Рисунок 2.17 – Прототип страницы с меню администратора

При нажатии на кнопку «Статистика» открывается окно со статистикой. Прототип страницы со статистикой представлен на рисунке 2.18.

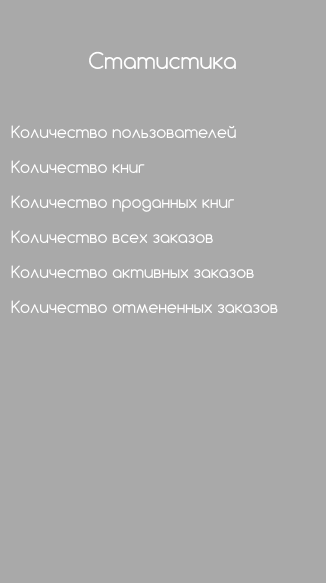


Рисунок 2.18 – Прототип страницы со статистикой

При нажатии на кнопку «Заказы» открывается страница со всеми заказами. Прототип страницы со статистикой представлен на рисунке 2.19.

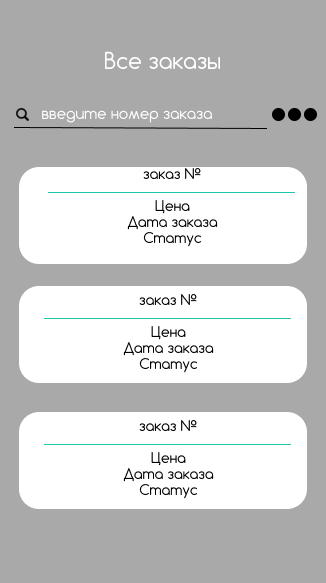


Рисунок 2.19 – Прототип страницы со всеми заказами

При нажатии на определенный заказ открывается страница с информацией о заказе, где можно менять статус заказа. Прототип страницы с информацией о заказе представлен на рисунке 2.20.

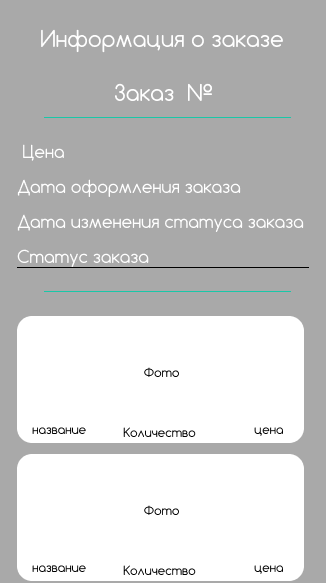


Рисунок 2.20 – Прототип страницы с информацией о заказе

При нажатии на кнопку «Все книги» открывается страница со списком всех книг. Прототип со списком всех книг страницы представлен на рисунке 2.21.



Рисунок 2.21 – Прототип страницы со списком всех книг

При нажатии на определённую книгу открывается информация о книге, где можно изменять количество книги. Прототип страницы с информацией о книге представлен на рисунке 2.22.



Рисунок 2.22 – Прототип страницы с информацией о книге

При нажатии на кнопку «Добавить книгу» открывается окно с полями для добавления новой книги. Прототип страницы с полями для добавления новой книги представлен на рисунке 2.23.



Рисунок 2.23 – Прототип страницы с полями для добавления новой книги

При нажатии на кнопку «Отзывы» открывается страница со всеми отзывами, где можно их удалять. Прототип страницы с отзывами представлен на рисунке 2.24.

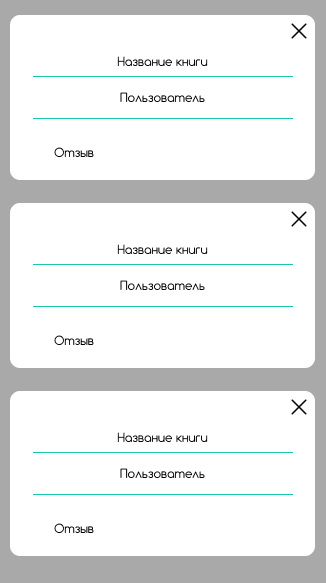


Рисунок 2.24 – Прототип страницы с отзывами

При нажатии на кнопку «Поддержка» открывается окно со списком всех активных вопросов в поддержку. Прототип страницы с вопросами в поддержку представлен на рисунке 2.25.

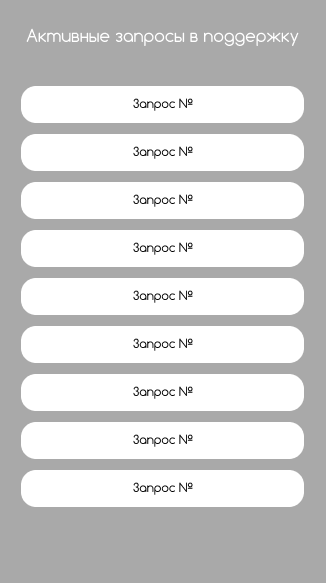


Рисунок 2.25 – Прототип страницы с вопросами в поддержку

При нажатии на определенный запрос в поддержку открывается страница с вопросом и полем ввода для ответа на вопрос. Прототип страницы с вопросом представлен на рисунке 2.26.



Рисунок 2.26 – Прототип страницы с вопросом

При нажатии на кнопку «Отчёты» открывается окно, где можно сформировать файл отчёта. Прототип страницы для формирования отчетов представлен на рисунке 2.27.

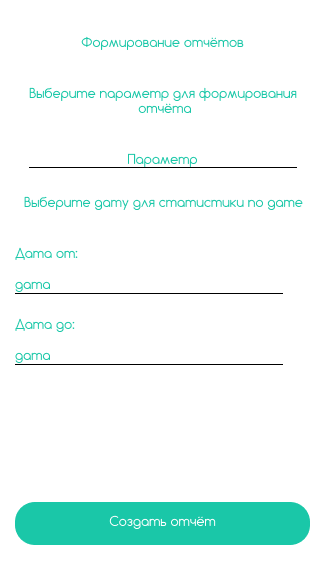


Рисунок 2.27 – Прототип страница для формирования отчетов

* 1. Защита и сохранность данных

Для мобильного приложения определены две роли – «Администратор» и «Покупатель». Роль «Администратор» будет обладать возможностями модерирования заказов, добавление книг, модерирование за отзывами, просмотр статистики, изменение количества книг. Роль «Покупатель» будет иметь все возможности, кроме возможностей роли «Администратор».

После создания ролей и назначения необходимых привилегий, следующим шагом будет создание пользователей и назначение им ролей. Для роли «Администратор» будет использоваться имя пользователя «admin» и пароль «1111». Для роли «Покупатель» будет использоваться логин «egorchuzhavko@gmail.com» и пароль «Qwerty!1».

Пользователям должны быть предоставлены только необходимые привилегии для обеспечения безопасности системы. Обычные пользователи базы данных не должны иметь доступа к системному каталогу или возможности изменять структуру базы данных.

* 1. Организация и ведение информационной базы (модели)

Цель моделирования данных состоит в обеспечении разработчика информационной системы концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему баз данных.

Наиболее распространенным средством моделирования данных являются диаграммы «сущность-связь» (ERD). С их помощью определяются важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных. Нотация ERD была впервые введена П. Ченном и получила дальнейшее развитие в работах Баркера. Диаграмма «сущность-связь» представлена на рисунке 2.28.

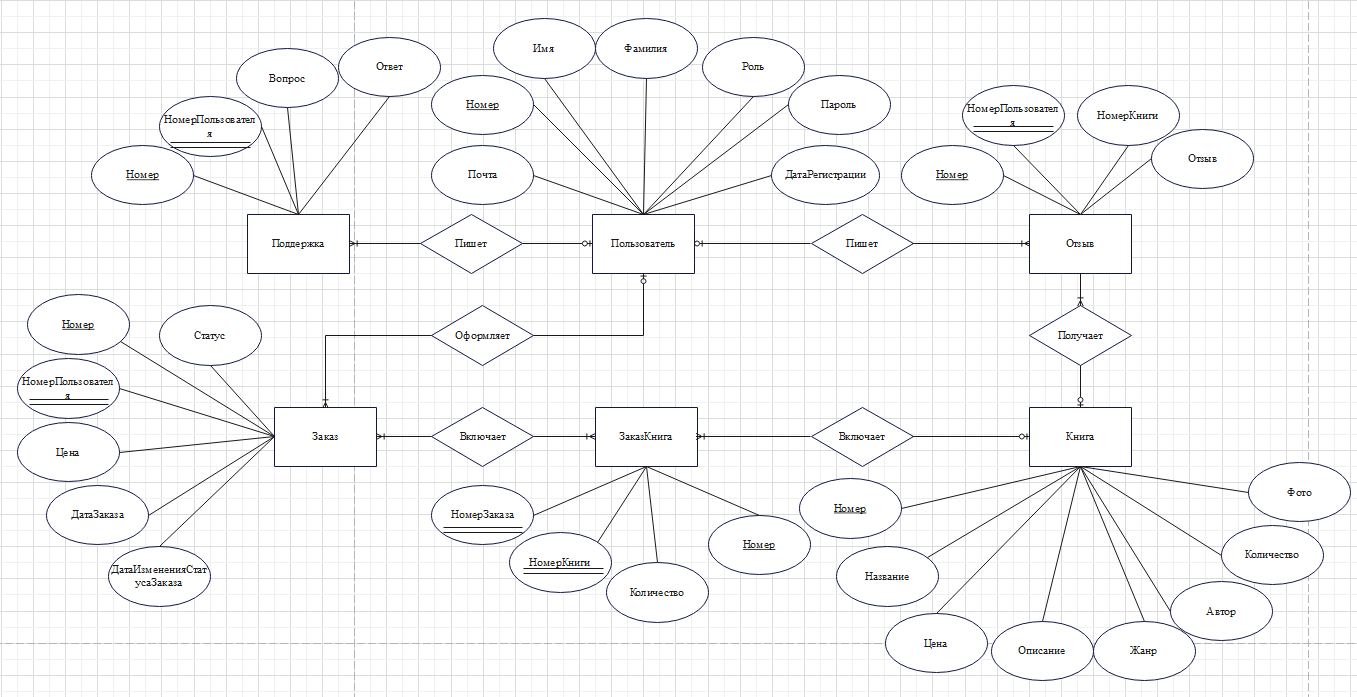


Рисунок 2.28 – Диаграмма «Сущность-связь»

Исходя из исследования предметной области, можно выделить следующие сущности разработки: «Пользователь», «Книга», «Заказ», «ЗаказКнига», «Отзыв», «Поддержка».

Для сущности «Пользователь» можно выделить следующие атрибуты:

* «Номер»;
* «Имя»;
* «Фамилия»;
* «Почта»;
* «Пароль»;
* «Роль»;
* «ДатаРегистрации».

Для сущности «Книга» можно выделить следующие атрибуты:

* «Номер»;
* «Название»;
* «Цена»;
* «Описание»;
* «Жанр»;
* «Количество»;
* «Фото»;
* «Автор».

Для сущности «Заказ» можно выделить следующие атрибуты:

* «Номер»;
* «Статус»;
* «НомерПользователя»;
* «Цена»;
* «ДатаЗаказа»;
* «ДатаИзмененияСтатусаЗаказа».

Для сущности «ЗаказКнига» можно выделить следующие атрибуты:

* «Номер»;
* «НомерЗаказа»;
* «НомерКниги»;
* «Количество».

Для «Отзыв» можно выделить следующие атрибуты:

* «Номер»;
* «НомерПользователя»;
* «НомерКниги»;
* «Отзыв».

Для «Поддержка» можно выделить следующие атрибуты:

* «Номер»;
* «НомерПользователя»;
* «Вопрос»;
* «Ответ».

Данное приложение было написано с использованием системы управления реляционными базами данных MySQL. Данная СУБД подходит для самых различных проектов: от небольших приложений до больших высоконагруженных проектов.

Для организации баз данных MySQL использует реляционную модель.

Таблица «Пользователь» хранит информацию о данных пользователя. Структура приведена в таблице 2.1.

Таблица 2.1 – Структура таблицы «Пользователь»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| Номер | Int | 4 | Уникальный ключ пользователя |
| Имя | Text | 65535 | Имя пользователя |
| Фамилия | Text | 65535 | Фамилия пользователя |
| Почта | Text | 65535 | Почта пользователя |
| Пароль | Text | 65535 | Пароль пользователя |
| Роль | Text | 65535 | Роль пользователя |
| ДатаРегистрации | DateTime | 8 | Дата регистрации |

Таблица «Заказ» хранит информацию о заказах. Структура приведена в таблице 2.2.

Таблица 2.2 – Структура таблицы «Заказ»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| Номер | Int | 4 | Уникальный ключ заказа |
| НомерПользователя | Int | 4 | Внешний ключ, указывающий на пользователя |
| Цена | Float | 4 | Цена заказа |
| ДатаЗаказа | DateTime | 8 | Дата, когда был оформлен заказ |
| ДатаИзмененияСтатусаЗаказа | DateTime | 8 | Дата, когда был изменен статус заказа |
| Статус | Text | 65535 | Активный статус заказа |

Таблица «ЗаказКнига» хранит информацию о книгах, которые хранятся в определенном заказе. Структура приведена в таблице 2.3.

Таблица 2.3 – Структура таблицы «ЗаказКнига»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| Номер | Int | 4 | Первичный ключ |
| НомерЗаказа | Int | 4 | Внешний ключ, указывающий на заказ |
| НомерКниги | Int | 4 | Внешний ключ, указывающий на книгу |
| КоличествоКниг | Int | 4 | Количество книг в заказе |

Таблица «Книга» хранит информацию о книгах. Структура приведена в таблице 2.4.

Таблица 2.4 – Структура таблицы «Книга»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| Номер | Int | 4 | Уникальный ключ книги |
| Название | Text | 65535 | Название книги |
| Цена | Float | 4 | Цена книги |
| Описание | Text | 65535 | Описание книги |
| Фото | Text | 65535 | Фото книги |
| Количество | Int | 4 | Количество оставшихся книг в наличии |

Таблица «Отзыв» хранит отзывы о книгах. Структура приведена в таблице 2.5.

Таблица 2.5 – Структура таблицы «Отзыв»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| Номер | Int | 4 | Первичный ключ |
| НомерПользователя | Int | 4 | Внешний ключ, указывающий на пользователя |
| НомерКниги | Int | 4 | Внешний ключ, указывающий на книгу |
| Отзыв | Text | 65535 | Отзыв на книгу |

Таблица «Поддержка» хранит вопросы, которые задаются в службу поддержки. Структура приведена в таблице 2.6.

Таблица 2.6 – Структура таблицы «Поддержка»

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Размер, байт | Описание |
| Номер | Int | 4 | Первичный ключ |
| НомерПользователя | Int | 4 | Внешний ключ, указывающий на пользователя |
| Вопрос | Text | 65535 | Вопрос, который задается |
| Отзыв | Text | 65535 | Ответ на вопрос |

1. Реализация мобильного приложения
   1. Программно-технические средства, необходимые для разработки приложения

Данный подраздел является важнейшим разделом документации любого проекта по разработке программного обеспечения. В нем описываются технологии и инструменты, использованные в процессе разработки, и дается подробный обзор их назначения и применения.

В данном конкретном случае программа была написана на языке программирования C# в среде разработки Visual Studio 2022, с использованием Xamarin Forms и базы данных MySQL.

Visual Studio 2022 – это интегрированная среда разработки, которая предлагает полный набор инструментов разработки для создания программных приложений с использованием языка C#. Она пользуется популярностью среди разработчиков благодаря удобному интерфейсу, мощным возможностям отладки и бесшовной интеграции с другими продуктами и сервисами Microsoft [6].

Xamarin – это инструмент для создания кроссплатформенных мобильных приложений с использованием C# и .NET. Он позволяет разработчикам создавать мобильные приложения, которые могут работать на разных платформах, таких как iOS, Android и Windows Phone. Это позволяет сэкономить время и ресурсы, которые могут быть затрачены на разработку отдельных приложений для каждой платформы [5].

База данных MySQL использовалась для хранения и управления данными, используемыми приложением.

PhpMyAdmin – веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL. PhpMyAdmin позволяет через браузер и не только осуществлять администрирование сервера MySQL, запускать команды SQL и просматривать содержимое таблиц и баз данных. Приложение пользуется большой популярностью у веб-разработчиков, так как позволяет управлять СУБД MySQL без непосредственного ввода SQL команд [10].

В целом, в подразделе представлен краткий обзор инструментов и технологий разработки, используемых в проекте. Подчеркивается важность выбора подходящих технологий для конкретной задачи и демонстрируется, как Visual Studio 2022, Xamarin и база данных MySQL были использованы для реализации проекта. Приведенные примеры демонстрируют универсальность инструмента разработки и его широкое применение в различных отраслях промышленности и секторах.

* 1. Описание разделов приложения

При входе в приложение пользователь видит только кнопки для входа и регистрации. После входа в аккаунт открывается экран, который зависит от роли пользователя.

Если пользователь является покупателем, то откроется экран с навигационным меню. В навигационном меню есть 3 раздела. Первый раздел представляет собой каталог книг, в котором представлены доступные книги для пользователя. При переходе к определённой книге пользователь может посмотреть подробную информацию о книге, а также посмотреть отзывы о книге. Второй раздел представляет собой корзину товаров, где показаны выбранные книги, которые планирует пользователь заказать. Можно изменить количество книг, либо удалить книгу. Также можно просмотреть информацию о книге и отзывы о ней. Третий раздел представляет собой профиль пользователя. Можно открыть информацию о самом пользователе и её изменить, просмотреть запросы в службу поддержки, просмотреть все заказы, либо активные заказы.

Если пользователь является администратором, то после входа ему откроется экран с панелью администратора. Администратору представляется возможность просмотра статистики, просмотр заказов, где можно просмотреть информацию о каждом заказе и изменить статус заказа, просмотр книг и изменение их количества, добавление книг, что представляет собой ввод данных о новой добавляемой книге и добавлением фотографии книги, отзывы, где можно удалить отзывы, которые можно назвать недопустимыми и поддержка, что представляет собой возможность ответить на вопрос, который был задан покупателем администрации приложения.

* 1. Описание используемых функций и процедур

Данный подраздел играет важную роль в отражении структуры пользовательских функций и их реализации в базе данных. Он дает представление о взаимосвязи между функциями и различными объектами в базе данных. В простейшем случае эти объекты могут включать представления, индексы, хранимые процедуры, определяемые пользователем функции и триггеры.

Для данной базы данных была разработана процедура «ВсеЗаказыДляПользователя». Она используется для нахождения всех заказов для конкретного пользователя. В качестве входного параметра функция принимает идентификационный номер пользователя, чтобы по нему найти все заказы пользователя.

Код процедуры представлен ниже.

CREATE PROCEDURE `ВсеЗаказыДляПользователя` (IN ` idcust` int) NOT DETERMINISTIC NO SQL SQL SECURITY DEFINER

BEGIN

SELECT \* from Заказ WHERE Заказ.Номер=idcust;

END

Также была создана функция «УдалитьПользователя», которая удаляет всю связанную информацию с пользователем из базы данных. В качестве входного параметра функция принимает идентификационный номер пользователя, чтобы по нему найти и удалить всю связанную информацию.

CREATE FUNCTION ` УдалитьПользователя `(` iduser ` INT) RETURNS TINYINT(1) NOT DETERMINISTIC NO SQL SQL SECURITY DEFINER

BEGIN

DELETE from Заказ WHERE Заказ.НомерПользователя=iduser;

DELETE from Пользователь WHERE Пользователь.Номер=iduser;

RETURN TRUE;

END

Полный текст программных модулей мобильного приложения представлен в приложении А.

* 1. Функциональное тестирование

В процессе написания программного средства необходимо производить тестирование на правильность работы приложения. Одной из основных задач тестирования является устранение ошибок, происходящих при вводе данных.

Функциональное тестирование – это тестирование функций приложения на соответствие требованиям. Оно нужно для того, чтобы убедиться, что продукт работает правильно, выполняет все необходимые функции и соответствует требованиям заказчика. Ключевая задача функционального тестирования – это проверка функциональности продукта на соответствие требованиям. Оценка производится в соответствии с ожидаемыми и полученными результатами (на основании функциональной спецификации), при условии, что функции отрабатывали на различных значениях. Функциональное тестирование также помогает улучшить качество продукта и повысить удовлетворенность пользователей. Когда продукт работает правильно и соответствует требованиям, пользователи могут использовать его с уверенностью и доверием. Кроме того, функциональное тестирование позволяет сэкономить время и ресурсы на исправление дефектов и улучшение продукта в дальнейшем.

В таблице 3.1 представлен тест-кейс для тестирования функции регистрации нового пользователя.

Таблица 3.1 – Test-case для тестирования функции регистрации нового пользователя

|  |  |  |
| --- | --- | --- |
| Описание | Шаги воспроизведения | Результат |
| Регистрация незарегистрированного пользователя, ввод корректных данных | 1.Войти в приложение;  2.Нажать на кнопку «Нет аккаунта? зарегистрироваться»;  3.Заполнить поля:  Имя – Егор  Фамилия – Чужавко  Почта – emailemail@gmail.com  Пароль – Qwerty!1  Повторите пароль – Qwerty!1;  4.Нажать на кнопку «Зарегистрироваться». | Ожидаемый результат: Сообщение об успешной регистрации |
| Фактический результат: результат соответствует ожидаемому. Все условия соблюдены.  Результат представлен в приложении В на рисунке Б.1. |

В таблице 3.2 представлен тест-кейс для тестирования функции регистрации нового пользователя с некорректными значениями.

Таблица 3.2 – Test-case для тестирования функции регистрации нового пользователя

|  |  |  |
| --- | --- | --- |
| Описание | Шаги воспроизведения | Результат |
| Регистрация незарегистрированного пользователя, ввод некорректных данных | 1.Войти в приложение;  2.Нажать на кнопку «Нет аккаунта? зарегистрироваться»;  3.Заполнить поля:  Имя – Егор  Фамилия – Чужавко  Почта – emailemail@gmail.com  Пароль – qwerty11  Повторите пароль – qwerty11;  4.Нажать на кнопку «Зарегистрироваться». | Ожидаемый результат: Сообщение об ошибке регистрации |
| Фактический результат: результат соответствует ожидаемому. Все условия соблюдены. Результат представлен в приложении Б на рисунке Б.2. |

В таблице 3.3 представлен тест-кейс для авторизации уже существующего пользователя с верными данными.

Таблица 3.3 – Test-case для тестирования функции авторизации существующего пользователя

|  |  |  |
| --- | --- | --- |
| Описание | Шаги воспроизведения | Результат |
| Авторизация существующего пользователя, ввод корректных данных | 1.Войти в приложение;  2.Нажать на кнопку «Войти»;  3.Заполнить поля: Почта – emailemail@gmail.com  Пароль – Qwerty!1  4.Нажать на кнопку «Войти». | Фактический результат: результат соответствует ожидаемому. |
| Все условия соблюдены. Результат представлен в приложении Б на рисунке Б.3. |

В таблице 3.4 представлен тест-кейс для тестирования функции оформления заказа.

Таблица 3.4 – Test-case для тестирования функции оформления заказа

|  |  |  |
| --- | --- | --- |
| Описание | Шаги воспроизведения | Результат |
| Оформление заказа | 1.Войти в приложение;  2.Нажать на кнопку «Войти»;  3.Заполнить поля: Почта – emailemail@gmail.com  Пароль – Qwerty!1  4.Нажать на кнопку «Войти».  5.Выбрать товар.  6.Нажать на кнопку «Добавить в корзину»  7.Перейти во вкладку «Корзина»  8.Нажать на кнопку «Оформить заказ»  9.Нажать на кнопку «Заказать» | Ожидаемый результат: Заказ был оформлен |
| Фактический результат: результат соответствует ожидаемому. Все условия соблюдены. |

В таблице 3.5 представлен тест-кейс для авторизации уже существующего пользователя с некорректными данными.

Таблица 3.5 – Test-case для авторизации существующего пользователя

|  |  |  |
| --- | --- | --- |
| Описание | Шаги воспроизведения | Результат |
| Авторизация существующего пользователя, ввод некорректных данных | 1.Войти в приложение;  2.Нажать на кнопку «Войти»;  3.Заполнить поля: Почта – emailemail@gmail.com  Пароль – Qwer  4.Нажать на кнопку «Войти». | Ожидаемый результат: Сообщение об ошибке авторизации |
| Фактический результат: результат соответствует ожидаемому. Все условия соблюдены. Результат представлен в приложении Б на рисунке Б.4. |

В таблице 3.6 представлен тест-кейс для тестирования функции добавления товара в корзину.

Таблица 3.6 – Test-case для тестирования функции добавления товара в корзину

|  |  |  |
| --- | --- | --- |
| Описание | Шаги воспроизведения | Результат |
| Добавление товара в корзину | 1.Войти в приложение;  2.Нажать на кнопку «Войти»;  3.Заполнить поля: Почта – emailemail@gmail.com  Пароль – Qwerty!1  4.Нажать на кнопку «Войти».  5.Выбрать товар.  6.Нажать на кнопку «Добавить в корзину» | Ожидаемый результат: Товар был добавлен в корзину |
| Фактический результат: результат соответствует ожидаемому. Все условия соблюдены |

1. Применение
   1. Общие сведения

Основной целью мобильного приложения является автоматизация учета продаж книжной продукции. Приложением может пользоваться как покупатель, так и администратор приложения. Само приложение может использоваться для поиска книг, оформления заказов, слежения за заказами, написания отзывов к книгам, написания вопросов в службу поддержки, просмотра статистики, формирования отчетов, ведения учета книг и заказов, управления отзывами и написания ответов на вопросы в службе поддержки. Приложение создавалось целенаправленно на мобильные устройства для того, чтобы можно было пользоваться приложением в любом удобном месте. Удобство заключается в том, что приложение всегда доступно под рукой.

Установка мобильного приложения осуществляется при помощи Android Application Package файла, при запуске которого начинается непосредственно сама установка приложения. Нужно загрузить «BookShop.apk» на телефон, включить разрешение на установку неизвестных приложений, запустить сам файл установки.

После установки приложения появится иконка приложения на экране, которую необходимо запустить. Важно понимать, что для работы с приложением нужен постоянный доступ в интернет.

Для входа как пользователь нужно зарегистрироваться в приложении либо войти в существующий аккаунт.

Для входа как администратор необходимо при входе в аккаунт ввести «admin» в поле «Почта» и «1111» в поле «Пароль».

* 1. Назначение программного средства

Основным назначением мобильного приложения является облегчение ведения учета товаров и заказов. Также мобильное приложение позволяет упростить администраторам приложения слежение за товарами, добавление товаров, контроль над отзывами и над книгами, просмотр статистики, ответы на вопросы в поддержку и формирование отчётов. Также мобильное приложение предоставляет возможность просмотра каталога, оформления заказов в лице покупателя. Так как это реализация для мобильного устройства, то это делает процесс заказа книг более удобным, так как мобильные устройства с доступом в интернет используют большинство людей.

Потенциальной аудиторией приложения являются люди, которые интересуются каталогом книг, платежеспособные потенциальные покупатели, работники организации.

Данные пользователей защищены, так как они находятся на удаленном сервере, доступ к которому имеется только у работников организации. Пользователи со стороны организации не имеют возможности манипулировать данными пользователей, они могут только просмотреть информацию.

Покупатели могут искать, смотреть, и заказывать товары. Администраторы могут следить за заказами, статистикой, книгами. Также администраторы могут добавлять книги.

1. Охрана труда и окружающей среды
   1. Правовые, нормативные, социально-экономические и организационные вопросы охраны труда.

ООО "АПВБ" разработало и внедрило систему управления охраной труда (СУОТ), которая соответствует требованиям нормативных актов Республики Беларусь в области охраны труда и стандарта СТБ ISO 45001-2020 «Системы менеджмента здоровья и безопасности при профессиональной деятельности. Требования и руководство по применению». СУОТ направлена на обеспечение безопасных условий работы на предприятии, сохранение жизни и здоровья сотрудников в процессе выполнения их трудовых обязанностей, а также соблюдение законодательства в области охраны труда.

Для организации работы и контроля по охране труда на предприятии создана служба охраны труда, возглавляемая инженером по охране труда. Основными задачами службы охраны труда являются:

* координация деятельности отделов и подразделений предприятия по обеспечению здоровых и безопасных условий труда;
* совершенствование СУОТ;
* внедрение передового опыта и научных разработок в области безопасности и гигиены труда, распространение знаний об охране труда;
* информирование и консультирование работников предприятия, включая руководство, по вопросам охраны труда;
* контроль за соблюдением требований законодательства, технических нормативных правовых актов и локальных нормативных правовых актов, содержащих требования охраны труда.

Инженер ОТ организует выполнение мероприятий по устранению недостатков и нарушений в области охраны труда, выявленных в ходе ежемесячного контроля.

Организацию, оперативное руководство и координацию работы по охране труда, а также общее руководство управлением охраной труда осуществляет главный инженер ООО «АПВБ». Ответственным за реализацию функций СУОТ является инженер ОТ.

Функции инженера по охране труда:

* анализирует причины производственного травматизма и профессиональной заболеваемости на предприятии, разрабатывает меры по их профилактике, контролирует их выполнение и эффективность;
* участвует в организации разработки и реализации программ улучшения условий и охраны труда на предприятии, согласовывает их с заинтересованными структурными подразделениями и органами государственного надзора;
* организует обучение, повышение квалификации и проверку знаний по вопросам охраны труда руководителей и специалистов предприятия, проводит инструктажи, лекции, тренинги, консультации по актуальным темам охраны труда;
* организует и руководит работой кабинета охраны труда в соответствии с Типовым положением о кабинете охраны труда, утвержденным постановлением Министерства труда Республики Беларусь от 29.12.2006 № 204, обеспечивает его материально-техническое оснащение, информационную поддержку, ведение документации;
* подготавливает статистическую отчетность по охране труда в соответствии с установленными формами и сроками, анализирует показатели по охране труда на предприятии, выявляет проблемные зоны и направления улучшения.

В соответствии с Законом Республики Беларусь от 23.06.2008 № 356-З «Об охране труда» и Трудовым кодексом Республики Беларусь, работодатель обязан осуществлять обучение, стажировку, инструктаж и проверку знаний работников по вопросам охраны труда.

Обучение по охране труда работников предприятия проводится в соответствии с действующими Государственными стандартами, межотраслевыми положениями, типовыми инструкциями и программами.

Обучение руководителей и специалистов осуществляется по учебным планам и программам, разработанным на основе типового перечня вопросов для обучения и проверки знаний по вопросам охраны труда.

Руководитель предприятия утверждает список должностей специалистов, которым требуется прохождение стажировки, и определяет продолжительность стажировки в соответствии с требованиями законодательства в области охраны труда.

Проверка знаний руководителей и специалистов по вопросам охраны труда проводится не позднее месяца со дня назначения на должность, а также периодически согласно требованиям нормативных актов, но не реже одного раза в три года. Эта проверка осуществляется в специальных комиссиях, ответственных за проверку знаний в области охраны труда.

Инженер несет ответственность за организацию своевременного и качественного проведения обучения, инструктажей, стажировок и проверок знаний сотрудников в области охраны труда на предприятии. В структурных подразделениях эта ответственность возлагается на их руководителей.

По характеру и времени проведения инструктажи по охране труда подразделяются на:

* вводный проводят со всеми работниками, которые впервые принимаются на постоянную или временную работу, независимо от их образования, трудового стажа, стажа работы по этой профессии, специальности, должности, а также с командированными при участии их в производственном процессе или выполнении или выполнении работ на территории организации, учащимися и студентами, прибывшими на производственное обучение или практику. Проводит инженер по охране труда либо специалист, на которого возложены эти обязанности, а с учениками в учебном заведении – преподаватель либо мастер производственного обучения;
* первичный на рабочем месте проводится до начала производственной деятельности с работниками, принятыми на работу. Проводится с каждым работником индивидуально, с практическим показом безопасных приемов и методов работы;
* повторный проходят все работники независимо от квалификации, образования, стажа и характера выполняемой работы не реже одного раза в полугодие. Он проводится либо с группой работников, обслуживающих однотипное оборудование и в пределах общего рабочего места по программе первичного инструктажа на рабочем месте, либо в объеме инструкций по охране труда на рабочем месте;
* внеплановый проводится при введении в действие новых либо переработанных нормативных актов по охране труда или внесении изменений в них, а также при изменении технологического процесса, замене или модернизации оборудования, приборов и инструментов, сырья, материалов и иных факторов, воздействующих на охрану труда. Проводится индивидуально либо с группой работников одной профессии;
* целевой проводится при ликвидации последствий аварий, стихийных бедствий и катастроф, а также при производстве работ, на которые оформляется наряд-допуск или разрешение. Фиксируется в наряде-допуске, разрешении и ином документе.

Порядок расследования несчастных случаев на производстве определен Правилами расследования несчастных случаев на производстве и профессиональных заболеваний, утвержденными постановлением Совета Министров Республики Беларусь от 15 января 2004 г. № 30.

Целями расследования несчастных случаев являются:

* установление причин, вызвавших несчастный случай;
* установление виновного;
* разработка плана мероприятий, направленных на предотвращение подобных случаев.

Основными видами контроля являются:

* контроль за соблюдением законодательства об охране труда, осуществляемый руководителями и специалистами организации в соответствии с их должностными обязанностями;
* общественный контроль за соблюдением законодательства об охране труда, осуществляемый профсоюзами;
* периодический контроль за соблюдением законодательства об охране труда, осуществляемый представителями нанимателя с участием общественных инспекторов профсоюзов по охране труда (уполномоченных лиц по охране труда работников).

Периодический контроль функционирования СУОТ проводится нескольких видов, а именно:

* ежедневный – осуществляется в структурных подразделениях их руководителями с участием общественного инспектора профсоюза по охране труда;
* ежемесячный – осуществляется в отделах главой отдела с участием общественного инспектора профсоюза по охране труда, руководителей отдела и ответственных за охрану труда;
* ежеквартальный – осуществляется в организации в целом руководителем организации с участием руководителей служб общественного инспектора профсоюза по охране труда.
  1. Анализ метеорологических условий на рабочем месте

Анализ метеорологических условий на рабочем месте является важным аспектом охраны труда, поскольку погодные условия могут оказывать негативное влияние на здоровье и безопасность работников. Для проведения анализа метеорологических условий необходимо учитывать следующие факторы:

* температура воздуха: высокая температура может привести к перегреву и дегидратации работников, а низкая температура может вызвать обморожение. Также следует учитывать влажность воздуха и скорость его движения;
* освещенность – недостаточное освещение может привести к напряжению глаз и усталости, а слишком яркое освещение может вызвать ослепление и раздражение глаз;
* атмосферное давление – сильные изменения атмосферного давления могут вызвать головные боли и ухудшение самочувствия у некоторых людей;
* атмосферные осадки – дождь, снег, град и другие атмосферные осадки могут вызвать скольжение на полу и создать опасность для работников;
* уровень шума – сильный шум на рабочем месте может привести к потере слуха и другим заболеваниям уха;
* качество воздуха – наличие вредных веществ в воздухе на рабочем месте может привести к отравлению и другим заболеваниям.

Для анализа метеорологических условий на рабочем месте необходимо проводить регулярные измерения температуры, влажности, скорости воздуха, уровня шума и других параметров, а также обеспечивать своевременную вентиляцию, освещение и другие условия, необходимые для безопасной работы. При необходимости следует принимать соответствующие меры для устранения негативных факторов влияния метеорологических условий на рабочем месте.

Для проведения анализа метеорологических условий на рабочем месте необходимо использовать соответствующие приборы и инструменты, такие как термометры, гигрометры, анемометры, звукомеры и другие. Измерения необходимо проводить регулярно, в соответствии с установленным графиком и требованиями законодательства.

При проведении анализа метеорологических условий необходимо учитывать специфику выполняемых работ и условия, в которых они происходят. Например, для работ, связанных с высокой температурой и влажностью, необходимо обеспечить доступ к прохладной воде и отдыху, а для работ в холодных условиях – обеспечить теплую одежду и обогревательные устройства.

Также необходимо обеспечить своевременную вентиляцию рабочих помещений, особенно в случаях, когда выполняются работы, связанные с выделением вредных веществ. Для этого могут использоваться системы приточно-вытяжной вентиляции, кондиционеры и другие устройства.

Освещенность на рабочем месте также является важным фактором охраны труда. Недостаточное освещение может привести к ухудшению зрения и увеличению вероятности ошибок при работе, а слишком яркое освещение может вызвать ослепление и раздражение глаз. Для обеспечения оптимальной освещенности необходимо использовать специальные лампы и осветительные приборы, а также учитывать рефлексию света от поверхностей в помещении.

Атмосферное давление также может оказывать влияние на здоровье работников, особенно при резких изменениях давления. Для предотвращения негативного влияния необходимо следить за погодными условиями и принимать соответствующие меры при изменении атмосферного давления.

Уровень шума и качество воздуха на рабочем месте также могут оказывать влияние на здоровье работников. Для предотвращения негативного влияния необходимо использовать звукопоглощающие материалы и системы, которые уменьшают уровень шума в рабочем помещении. Также необходимо обеспечить постоянную проветриваемость помещения и установить систему очистки воздуха, если это необходимо.

После проведения анализа метеорологических условий необходимо принимать соответствующие меры по обеспечению безопасных и комфортных условий труда для работников. Эти меры могут включать в себя использование специальной защитной одежды, установку систем вентиляции и кондиционирования воздуха, улучшение освещения и другие.

Важно также проводить регулярный мониторинг метеорологических условий на рабочем месте, чтобы убедиться в эффективности принятых мер и своевременно выявлять возможные проблемы. Если выявляются какие-либо проблемы, необходимо принимать меры для их устранения, чтобы обеспечить безопасные и комфортные условия труда для всех работников.

После проведения анализа метеорологических условий на рабочем месте, возможно, потребуется внести изменения в условия труда и принять меры по улучшению охраны труда работников. Для этого могут быть разработаны соответствующие рекомендации, которые будут основаны на результатах анализа.

Рекомендации по улучшению метеорологических условий на рабочем месте могут включать в себя:

* обеспечение достаточной вентиляции в рабочем помещении. Если уровень воздухообмена недостаточен, необходимо установить систему приточно-вытяжной вентиляции, которая обеспечит свежий воздух и удаление загрязненного воздуха;
* использование системы кондиционирования воздуха в случаях, когда температура и влажность на рабочем месте слишком высоки или низки. Также необходимо обеспечить регулярное обслуживание и чистку кондиционеров для предотвращения возможной аллергической реакции у работников;
* использование специальных ламп и осветительных приборов для обеспечения достаточного уровня освещенности на рабочем месте. Необходимо учитывать рефлексию света от поверхностей и выбирать подходящие типы ламп в соответствии с видом работы, которая выполняется;
* обеспечение комфортной температуры на рабочем месте. В зависимости от выполняемой работы, необходимо поддерживать оптимальную температуру в помещении. Для этого могут использоваться обогреватели или кондиционеры;
* использование специальной защитной одежды и других средств защиты в случаях, когда это необходимо. Например, для работ, связанных с высокой температурой и влажностью, необходимо использовать специальную одежду и средства защиты;
* проведение обучения и тренингов для работников по охране труда и мерам предотвращения возможных проблем, связанных с метеорологическими условиями на рабочем месте;
* регулярный мониторинг метеорологических условий на рабочем месте и обновление рекомендаций в случае изменения климатических условий.

Важно отметить, что рекомендации по улучшению метеорологических условий на рабочем месте должны быть разработаны с учетом специфики выполняемой работы и индивидуальных особенностей работников. Кроме того, для успешной реализации рекомендаций необходимо обеспечить контроль их выполнения и регулярно оценивать эффективность принятых мер.

* 1. Пожарная безопасность

В соответствии с техническими кондиционами и Правилами (ТКП) 474-2013, помещение, в котором будет использоваться программный продукт, входит в категорию В1-В4 — пожароопасных помещений. В данном помещении присутствуют горючие вещества и материалы, которые могут возгораться при взаимодействии с кислородом или воздухом.

Согласно Правилам Устройства Электроустановок (ПУЭ), помещения классифицируются по зонам. Помещение, в котором создается дипломный проект, относится к зоне П-IIa, так как оно находится в помещении, где имеются твёрдые горючие вещества.

При пожаре строительные конструкции имеют следующие предел огнестойкости и класс пожарной опасности:

* несущие элементы здания (R 45-K1) – предел огнестойкости 45 минут - по потере несущей способности, мало пожароопасные;
* самонесущие стены (RE 30-K1) – предел огнестойкости 30 минут - по потере несущей способности и потере целостности независимо от того, какое из двух предельных состояний наступит ранее, мало пожароопасные;
* наружные несущие стены (E 15-K2) – предел огнестойкости 15 минут - по потере целостности, умеренно пожароопасные;
* перекрытия междуэтажные (REI 45-K1) – предел огнестойкости 45 минут - по потере несущей способности, целостности и теплоизолирующей способности независимо от того, какое из трех предельных состояний наступит ранее, мало пожароопасные;
* элементы покрытий без чердаков – настилы, в том числе с утеплителем (RE 15-K1) - предел огнестойкости 15 минут – по потере несущей способности и потере целостности независимо от того, какое из двух предельных состояний наступит ранее, мало пожароопасные;
* элементы покрытий без чердаков – фермы, балки, прогоны (R 15-K1) - предел огнестойкости 15 минут - по потере несущей способности, мало пожароопасные;
* лестничные клетки - внутренние стены (REI 60-KO) - предел огнестойкости 60 минут - по потере несущей способности, целостности и теплоизолирующей способности независимо от того, какое из тех предельных состояний наступит ранее, не пожароопасные;
* лестничные клетки - марши и площадки лестниц (R 45-KO) - предел огнестойкости 45 минут – по потере несущей способности, не пожароопасные.

Организация использует порошковые и углекислотные огнетушители марки ОП-5 и ОУ-10 соответственно в качестве первичных средств пожаротушения. Эти огнетушители выбраны, так как наиболее подходят для тушения электронной техники и близлежащих объектов. Также в организации имеются пожарные краны. Огнетушители размещены на видных и легкодоступных местах для сотрудников, чтобы быстро тушить возгорания, и они не препятствуют безопасной эвакуации людей. Расстояние от возможного очага пожара до места размещения первичных средств пожаротушения должно быть не более 20 метров. Пожарные краны марки Ду50 размещены на коридоре в специально оборудованных полках с закрывающейся дверью и снабжены пломбами.

В организации отсутствуют средства автоматического пожаротушения. Для оповещения о пожаре используются дымоуловители и пожарные извещатели. Дымоуловители равномерно расположены по помещениям и установлены на потолке. Пожарные извещатели размещены в помещениях на стенах в легко доступных местах.

На предприятии существует пожарная дружина, как требуется в соответствии с установленными нормами. Численный состав дружины составляет 10 человек, поскольку численность организации не превышает 100 человек. Командиры и старшие расчетов назначены руководителем организации. В помещениях на видном месте размещен табель действий при пожарах. Все члены дружины своевременно проходят контроль знаний.

Основными задачами пожарной дружины являются:

* контроль за соблюдением противопожарного режима;
* проведение разъяснительной работы среди работников по соблюдению противопожарного режима на рабочих местах и правил осторожного обращения с огнем в быту;
* надзор за исправностью средств пожаротушения и их укомплектованностью;
* вызов пожарной службы в случае возникновения пожара, принятие мер по его тушению имеющимися средствами и т.п.

Члены пожарной дружины принимают участие в локализации и ликвидации загораний, эвакуации людей и материальных ценностей из горящих помещений.

При эвакуации сотрудники покидают рабочие места, направляются к лестнице, по которой осуществляют спуск, затем выходят из здания. Пути эвакуации отражены на плане эвакуации. Сам план содержит схему занимаемых помещений организацией, с обозначениями. Каждый сотрудник обязательно ознакамливается с планом эвакуации.

* 1. Охрана окружающей среды

Согласно санитарным нормам, в помещении ООО «АПВБ» требуется обеспечить доступ к свежему воздуху. Это может быть достигнуто через форточное проветривание, кондиционирование или ионизацию воздуха.

В основном используется форточное проветривание в соответствии с установленным графиком. Согласно графику, помещение должно быть проветрено минимум два раза в день, и в это время сотрудники не должны присутствовать в помещении. Кондиционер используется для поддержания комфортной температуры в помещении.

План очистки воздуха:

* оповестить сотрудников в занимаемом помещении;
* открыть форточку в режим проветривания;
* освободить помещение;
* закрыть помещение на время проветривания.

В организации используемая вода, включая сточные воды, направляется в канализацию с целью последующей обработки в водоочистительных сооружениях. Важно отметить, что прием воды должен осуществляться в соответствии с требованиями местных исполнительных органов и взаимодействием с органами Министерства природных ресурсов и охраны окружающей среды Республики Беларусь. Эти требования обычно регулируются Правилами использования централизованных систем водоснабжения и водоотведения (канализации) в населенных пунктах. Способы очистки сточных вод:

* механический – размельчение осадков и отложений различными инструментами и механизмами;
* гидромеханический – подача струи воды под давлением для размыва и выноса отложений и грязи;
* физико-химический – циркуляция специально подобранных растворителей в аппаратуре без ее вскрытия;
* выжигание отложений в струе воздуха без вскрытия аппаратов и др.

Вывозом и утилизацией бытового мусора, занимается сторонняя организация «Remondis», согласно нормам, в существующем законодательстве. Сама утилизация производится термически. Термическая утилизация, подразумевает собой переработку отходов, путем сжигания. Способы утилизации бытового мусора:

* захоронение – производственный мусор, который не может быть использован вторично, подвергается хранению на специальных полигонах и свалках. Сбор, переработка и складирование радиоактивных отходов — большая проблема для любого государства. Их хранят в могильниках, которые должны быть расположены как можно дальше от грунтовых вод, в сейсмически безопасных зонах;
* компостирование – естественное разложение тяжёлых бытовых отходов. В результате процесса получают компост, который эффективно используется в сельском хозяйстве. В Беларуси способ утилизации не получил широкого распространения;
* термическая переработка – превращение мусора в топливо, путём его сжигания;
* плазменная переработка – процесс представляет собой разложение ТБО без доступа воздуха. Мусор нагревается и расщепляется на вещества с более низкой молекулярной массой. Пиролиз имеет больше преимуществ, чем сжигание: не загрязняется окружающая среда, тяжелые металлы превращаются в золу, в остатках отходов не содержится токсичных веществ.

В организации используются лампы, как люминесцентные, так и ртутные. По истечению срока использования ламп прибытию их в негодность, нужна их утилизация. Утилизация ламп осуществляется путем оформления заявки в организации «БелВторСырье», затем вывоз и утилизация.

1. Экономический раздел
   1. Технико-экономическое обоснование разработки программного средства

Вывод на рынок мобильного приложения по автоматизации учёта книжной продукции может иметь ряд преимуществ для бизнеса:

* удобство для пользователей: мобильное приложение позволяет пользователям легко получать доступ к информации о книгах, просматривать каталог и совершать покупки, вести учет книг и заказов, формирование отчетов, возможность писать вопросы в службу поддержки и возможность писать ответ на эти вопросы, добавление новых книг, где бы они ни находились;
* увеличение вовлеченности пользователей: мобильные приложения обычно предлагают удобный интерфейс, позволяющий пользователям быстро находить то, что ищут, и выполнять действия, такие как покупка книг, учет книг;
* расширение аудитории: мобильные приложения могут помочь бизнесу достичь новых клиентов, которые предпочитают использовать смартфоны и планшеты для покупок и поиска информации;
* повышение лояльности клиентов: мобильное приложение может включать такие функции как история покупок, написание отзывов к книгам и написание вопросов в поддержку, что помогает удерживать клиентов и увеличивать их лояльность;
* улучшение продаж: мобильное приложение может улучшить продажи, предоставляя пользователям легкую и удобную возможность просматривать и покупать книги;
* взаимодействие с клиентами: в мобильном приложении имеется поддержка, в которую можно писать будет писать пожелания по улучшению, либо найденные ошибки в программном средстве.

Цель экономического раздела дипломного проекта – рассчитать затраты на разработку программного продукта и определить экономическую эффективность от его внедрения.

Для этого необходимо решить ряд действий:

* составить план по разработке программного средства;
* рассчитать стоимость разработки;
* рассчитать экономическую эффективность разработки;
* рассчитать экономический эффект для свободной реализации на рынке.
  1. Составление плана по разработке программного продукта

План разработки программного средства представлен в таблице 6.1.

Таблица 6.1 - План разработки программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапов и видов работ | Исполнитель  (должность,  квалификация) | Количество исполнителей | Трудоемкость, человеко-дни, месяцы |
| Подготовительный | техник-программист | 1 | 2 |
| Выбор методов и средств | техник-программист | 1 | 2 |
| Разработка алгоритмов и программ | техник-программист | 1 | 22 |
| Отладка программ и анализ результатов | техник-программист | 1 | 10 |
| Оформление документации и подготовка к сдаче разработки | техник-программист | 1 | 4 |

Наиболее сложной и ответственной частью составления плана разработки программного продукта является расчет трудоемкости этапов и работ (графа 4 таблица 6.1).

Гистограмма, отображающая стадии разработки программного продукта приведена на рисунке 6.1.

Рисунок 6.1 – Гистограмма распределения времени работы при разработке программного продукта

* 1. Расчет затрат на разработку программного продукта

Стоимостная оценка программного продукта и определение экономического эффекта у разработчика предполагает расчет затрат, которые включают следующие элементы:

* материальные затраты;
* затраты на оплату труда;
* отчисления на социальные нужды;
* амортизация основных средств и нематериальных активов;
* прочие затраты.

Материальными затратами являются затраты на материалы и принадлежности, необходимые для проведения научно-исследовательских работ, магнитные носители, бумагу, красящие ленты и другие материалы, необходимые для разработки программного продукта. Затраты определяются по действующим отпускным ценам. Здесь так же отражаются затраты на электроэнергию.

Расчет осуществляется по формуле (6.1)

 (6.1)

где Ктр – коэффициент, учитывающий транспортно-заготовительные расходы (в работе принимаем от 1,05 до 1,10);

Нpi – норма расхода i-го вида материалов;

Цi –действующая отпускная цена за единицу i-го вида материала, руб.;

n – количество применяемых видов материалов.

Рм = 1,10 x (30,00 x 1 + 1,40 x 1 + 0,90 x 1 + 2,00 x 1) = 37,62 руб.

Расчет целесообразно представить в табличной форме (таблица 6.2).

Таблица 6.2 - Расчет материальных затрат­­­

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование материалов | Единица измерения | Количество | Цена приобретения, руб. | Сумма, руб. |
| Бумага | пачка | 1 | 30,00 | 30,00 |
| Шариковая ручка | штука | 1 | 1,40 | 1,40 |
| Папка-скоросшиватель | штука | 1 | 0,80 | 0,80 |
| Диск | штука | 1 | 2,00 | 2,00 |
| Всего расходов | | | | 34,20 |
| Всего с транспортно-заготовительными расходами | | | | 37,62 |

Затраты на электроэнергию находятся исходя из продолжительности периода разработки программного продукта, количества кВт/ч, затраченных на его проектирование и тарифа за 1 кВт/ч. по следующей формуле (6.2)

(6.2)

где Кэ – стоимость одного кВт/ч,руб;

Тр – количество кВт/ч.

Тарифы на электроэнергию применяются согласно приложению к Декларации «Об уровне тарифов на электроэнергию, отпускаемую РУП Электроэнергетики ГПО “Белэнерго” для юридических лиц и ИП» на соответствующий период времени, когда разрабатывается программный продукт. Базовый тариф для прочих потребителей с 01.01.2023 г. составляет 0,43912 руб. за 1 кВт/ч. Время реализации проекта 40 дней, среднее потребление энергии в месяц составляет 75 кВт/ч, то есть необходимое потребление энергии в дипломном проекте принимается 143 кВт/ч. Исходя из вышеизложенного получаем, что на электроэнергию было затрачено:

Рэ = 0,43912 x 143 кВт/ч. = 62,79 руб.

Величина затрат на оплату труда исчисляется исходя из численности различных категорий исполнителей и трудоемкости выполнения отдельных видов работ, тарифных ставок за один день или месячных должностных окладов, премиальных систем оплаты труда исполнителей по формуле (6.3)

(6.3)

где Tci – тарифная ставка за день (месячный оклад) i-й категорий работников;

tфi – время фактической работы работника i-й категории по теме, дн. или мес.;

– коэффициент премий по премиальным системам, от 1,10 до 1,30.

Роз = 1,10 x 835,20 = 918,72 руб.

Определяется основная заработная плата научно-технического персонала, непосредственно занятого выполнением работ.

К этой статье относятся основная заработная плата работников, а также премии, входящие в фонд заработной платы. Среднее количество рабочих дней в месяце равно 21, а средняя продолжительность рабочего дня составляет восемь часов. Следовательно, часовая заработная плата определяется делением размера оклада на количество рабочих часов в месяце (то есть на 168 часов).

Тарифная ставка первого разряда на предприятии составляет 216 руб. Трудоемкость определяется исходя из данных, представленных в таблице 6.1.

Произведение трудоемкости на сумму часовой заработной платы определяет затраты по зарплате для работника на все время разработки.

Расчет данных прямых расходов также целесообразно представить в табличной форме (таблица 6.3).

Таблица 6.3 - Расчет затрат на основную заработную плату персонала

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование категорий работников и должностей | Количество штатных единиц, чел. | Заработная плата за 1 день (месяц), руб. | Трудозатраты, дн. или мес. | Сумма, руб. |
| Техник-программист | 1 | 20,88 | 40 | 835,20 |
| Всего | | | | 835,20 |
| Всего с коэффициентом премий | | | | 918,72 |

Определяется дополнительная заработная плата исполнителей, включающая разнообразные предусмотренные трудовым законодательством выплаты, по формуле (6.4)

(6.4)

где Ндз – норматив дополнительной заработной платы, от 10 до 20%.

Рдз = 918,72 x 0,10 = 91,87 руб.

Отражается сумма обязательных страховых взносов, взносов на профессиональное пенсионное страхование в бюджет государственного внебюджетного фонда социальной защиты населения Республики Беларусь.

Рассчитываются отчисления органам социальной защиты по формуле (6.5)

(6.5)

где Нос – норма отчислений на социальную защиту, Нос=34%.

Рос = (853,47 + 85,35) x 0,34 = 319,20 руб.

Также рассчитываются отчисления на страхование от несчастных случаев на производстве и профессиональных заболеваний (Рстр) по ставке действующего законодательства (Нбгс = 0,3% - 1%). Для расчетов среднее значение Нбгс принимается равным 0,6%. Страховые отчисления рассчитываются по следующей формуле (6.6)

Рстр = (6.6)

Рстр = (853,47 + 85,35) x 0,006 = 5,63 руб.

Амортизация основных средств и нематериальных активов рассчитываются в установленном законодательством Республики Беларусь порядке, по одному из методов начисления амортизации (линейный, нелинейный, производительный), исходя из целесообразности его применения.

Порядок начисления амортизации по объектам в необоротных активах в Республике Беларусь регулируется Положением «О порядке начисления амортизации основных средств и нематериальных активов».

Амортизация начисляется на все основные средства и нематериальные активы, находящиеся на балансе организации и отраслей промышленности, независимо от характера их участия в производственном процессе.

Денежное выражение амортизации является амортизационным отчислением, которое входит в текущие затраты. Величина амортизационных отчислений определяется на основе норм амортизации.

Норма амортизации— это установленный размер амортизационных отчислений на полное восстановление, выраженное в процентах. Норма амортизации устанавливается на основе экономически целесообразного срока службы и должна обеспечить возмещение износа основных средств к моменту возможного их морального и физического износа и создать экономическую основу для их замены.

Для определения затрат по данному элементу будет использоваться линейный способ начисления амортизации. Нормативные сроки службы машин и оборудования составляют 5 лет.

Норма амортизации для линейного способа начисления вычисляется по формуле (6.7):

(6.7)

где Тс – срок службы оборудования, лет.

На = 0,2 x 100 = 20%

Для линейного способа начисления амортизационные отчисления равномерно распределены на весь период службы оборудования и вычисляются на один год. Так как разработка программного продукта длилась 40 дней, то сумма амортизационных отчислений (АО) за этот период высчитывается по следующей формуле (6.8)

(6.8)

Рао = (3200 x 20% x 40) / (12 x 30) = 71,11 руб.

Прочие затраты на конкретное программное обеспечение включают затраты на: арендную плату; вознаграждения за рационализаторские предложения и выплата авторских гонораров; затраты на гарантийный ремонт и обслуживание изделий;начисленные налоги, сборы (пошлины), платежи, включаемые в затраты на производство продукции (работ, услуг); связь; на оплату услуг рекламы и маркетинга; банков и небанковских кредитно-финансовых организаций и пр.

В расчетах принимаем условно размер прочих затрат равных 30-50% от суммы всех остальных затрат на разработку. Прочие затраты рассчитываются по следующей формуле (6.9)

(6.9)

где Нпр ≈ от 20 до 30%.

Рпр = 853,35 x 0,2 = 170,67 руб.

На основании полученных данных по отдельным статьям затрат рассчитывается общая сумма затрат на разработку программного продукта.

Общая сумма затрат по элементам на разработку программного продукта рассчитывается по формуле (6.10)

(6.10)

З = 37,62 + 62,79 + 918,72 + 91,87 + 343,60 + 6,06 + 58,89 + 183,74 = 1703,29 руб.

Результаты расчетов оформляются таблицей 6.4.

Таблица 6.4 - Расчет затрат на разработку программного продукта

|  |  |
| --- | --- |
| Элемент затрат | Затраты, руб. |
| Материальные затраты (Рм) | 37,62 |
| Электроэнергия (Рэ) | 62,79 |
| Затраты на оплату труда (Роз) | 918,72 |
| Дополнительная заработная плата (Рдз ) | 91,87 |
| Отчисления органам социальной защиты (Рос); | 343,60 |
| Отчисления на страхования от несчастных случаев (Рстр) | 6,06 |
| Амортизация (РАо) | 58,89 |
| Прочие затраты (Рпр) | 183,74 |
| Общая сумма затрат (З) | 1703,29 |

Таблицу 6.4 удобно представить в виде круговой диаграммы, отображающей элементы затрат программного продукта. Круговая диаграмма структуры затрат представлена на рисунке 6.2.

Рисунок 6.2 – Круговая диаграмма структуры затрат

На основании результатов таблицы можно прийти к выводу, что самыми большими затратами при разработке программного продукта являются затраты на зарплату, на отчисления органам социальной защиты и прочие затраты. Самой маленькой затратой являются отчисления на страхования от несчастных случаев.

* 1. Расчет экономического эффекта у разработчика и пользователя программного продукта

### Экономический эффект у разработчика программного продукта

Заказчик оплачивает разработчику всю сумму расходов по проекту, включая прибыль. После уплаты налогов из прибыли в распоряжении заказчика остается чистая прибыль от проекта. Ввиду того, что программное обеспечение разрабатывается для одного объекта, чистую прибыль можно считать в качестве экономического эффекта разработчика от реализованного программного продукта.

В дипломном проекте отпускная цена программного продукта, представляет собой не цену за единицу продукции, а цену проекта, за которую его можно продать и получить определённую выгоду.

Отпускная цена продукции формируется исходя из плановой себестоимости производства продукции, всех видов установленных налогов и прибыли, а также качества, потребительских свойств продукции и конъюнктуры рынка.

С учетом действующих в республике нормативных документов отпускная цена на продукцию рассчитывается по формуле (6.11)

(6.11)

где ОЦ – отпускная цена разработчика, руб.;

З – затраты на разработку, руб.;

П – прибыль, руб.

Прибыль рассчитывается следующей формуле (6.12)

(6.12)

где *R* – уровень рентабельности (10-30%), %.

П = 1703,29 x 0,2 = 340,66 руб.

ОЦ = 1703,29 + 340,66 = 2043,95 руб.

Стоимость проекта с учётом НДС, представляет собой сумму отпускной цены и налога на добавленную стоимость. НДС рассчитывается по следующей формуле (6.13)

(6.13)

где ставка НДС – 20%.

НДС = (1703,29 + 340,66) x 0,2 = 408,79 руб.

Отпускная цена с НДС рассчитывается по следующей формуле (6.14)

(6.14)

ОЦс ндс = 2043,95 + 408,79 = 2452,74 руб.

Таким образом, разработчик программного продукта может продать заказчику программный продукт, что покроет затраты и обеспечит прибыль за разработку проекта.

### Экономический эффект у пользователя программного продукта

Экономический эффект у пользователя программного продукта выражается в виде экономии трудовых, материальных и финансовых ресурсов, получаемой от:

* повышение производительности сервиса (увеличение числа выводимых за единицу времени документов, уменьшение среднего времени подготовки отчета и т.д.), что выражается в снижении трудоемкости выполнения операций, решении задач, подготовки данных, обработки информации и анализа результатов;
* сокращение затрат на оплату машинного времени и расходных материалов;
* повышения уровня сервиса (сокращение времени на устранение инцидентов);
* улучшения показателей основной деятельности предприятия в результате использования программного продукта и т.д.

Так же можно проводить сравнение затрат на производство по статьям затрат (если учащийся располагают такой информацией по изделию-аналогу). За базовый вариант следует принимать аналогичный программный продукт, используемое в действующей автоматизированной системе или ручной вариант, если автоматизация отсутствует.

При сравнении базового и нового варианта программного продукта в качестве экономического эффекта будет выступать общая сумма экономии всех видов ресурсов относительно базового варианта.

Приведем расчет экономии некоторых видов ресурсов в связи с использованием нового программного обеспечения.

Экономия затрат на заработную плату (Эз) при использовании нового программного продукта в расчете на объем выполненных работ определяется по формуле (6.14)

(6.14)

где Эзе – экономия затрат на заработную плату при решении задач с использованием нового программного продукта в расчете на 1 задачу, руб.

Q – количество задач, решаемых за год с использованием нового программного продукта (задач).

Эз = 2,09 x 1500 = 3135 руб.

Экономия затрат на заработную плату в расчете на 1 задачу (Эзе) рассчитывается по формуле (6.15)

(6.15)

где Зсм – среднемесячная заработная плата одного программиста, руб.;

Тс1, Тс2 – снижение трудоемкости на одну задачу в базовом и новом варианте соответственно, человеко-часов (соответственно 1,71 и 0,87 чел-час);

Тч – количество часов работы в день, часов (8 часов)

Др – среднемесячное количество рабочих дней.

Эзе = (417,60 x (1,71 - 0,87) / 8) / 21= 2,09 руб.

Экономия с учетом начислений на заработную плату (Эн) определяется по следующей формуле (6.16)

(6.16)

где Кнз – коэффициент начислений на заработную плату – 1.2.

Эн = 3135 x 1.2 = 3762 руб.

Экономия за счет сокращения простоя сервиса (Эс) определяется по следующей формуле (6.17)

(6.17)

где П1, П2 – время простоя сервиса, обусловленное программным продуктом в день;

Дрг – плановый фонд работы сервиса (дней);

Сп – стоимость одного часа работника для которого разрабатывается программный продукт, руб.

Эс = ((30 - 10) x 252 x 3,5) / 60 = 294 руб.

Общая экономия от применения нового программного продукта рассчитывается по формуле (6.18)

(6.18)

Эо = 3762 + 294 = 4056 руб.

Внедрение нового программного продукта позволит сэкономить на текущих затратах, т.е. практически получить на эту сумму дополнительную прибыль. Для пользователя в качестве экономического эффекта выступает лишь чистая прибыль – дополнительная прибыль, остающаяся в его распоряжении, которая определяется по формуле (6.19)

(6.19)

Пч = 4056 – (4056 x 20) / 100 = 3244,8 руб.

За счёт введения программного средства в эксплуатацию появляется дополнительная выручка предприятия, которая за год составила 5123.26 рублей. При дополнительной выручке предприятия появляется дополнительная прибыль, которая считается по следующей формуле (6.20)

(6.20)

где, Вндс – дополнительная выручка с НДС за год;

З – сумма всех затрат.

Вндс = 5123,26 + 5123,26 х 20% = 6147,91 руб.

Пд = 6147,91 - 2452,74 – ((6147,91 - 2452,74) х 20) / 100 = 2956,14 руб.

* 1. Расчет экономического эффекта для свободной реализации на рынке ИТ

В данном разделе представлено экономическое обоснование для дипломного проекта.

В технико-экономическом обосновании были рассмотрены следующие вопросы:

* составление плана по разработке программы;
* расчет стоимости разработки;
* расчет экономического эффекта у разработчика программного продукта;
* расчет экономического эффекта у пользователя программного продукта.

В первой части технико-экономического обоснования был рассчитан срок разработки по созданию программного средства. Он составил 320 часов.

Во второй части технико-экономического обоснования была рассчитана стоимость разработки программного продукта, которая составила 2452,74 рублей.

Дополнительная прибыль за счёт внедрения программного средства примерно составляет 2956,14 рублей.

Основное преимущество разработки программного средства в том, что оно позволяет пользователям заказывать в любое время и в любом месте любую интересующую его книгу, имея только телефон и доступ в интернет.

Заключение

В рамках дипломного проекта было разработано мобильное приложение для автоматизации учета книжной продукции, позволяющее оформлять заказы со стороны покупателя и вести учёт со стороны администратора.

В процессе работы был проведен анализ существующих аналогов программного обеспечения, программных средств и технологий для разработки мобильных приложений. Были созданы диаграммы, описывающие функциональную модель приложения, и описан процесс разработки графического интерфейса.

Для обеспечения надежности приложения была проведена опытная эксплуатация, которая подтвердила корректность всех функций приложения, отсутствие сбоев и неадекватных реакций программы. Приложение обладает несколькими преимуществами, такими как простой и понятный интерфейс, низкий объем памяти, быстрая обработка данных и отсутствие жестких требований к производительности и архитектуре системы.

Для подробного описания разработанного программного средства была составлена пояснительная записка, включающая в себя восемь разделов с информацией о мобильном приложении, включая раздел по охране труда и окружающей среды, а также экономический раздел.

В разделе охраны труда были рассмотрены такие вопросы, как правовые, нормативные, социально-экономические и организационные вопросы охраны труда, анализ метеорологических условий на рабочем месте, вопросы пожарной безопасности, вопросы охраны окружающей среды.

В экономическом разделе произведены расчеты экономического эффекта программного средства, расчеты заработной платы исполнителей, проведена оценка сложности и трудоемкости разработки программного средства, проведены расчеты всевозможных отчислений и расходов. Стоимость разработки программного средства составила 2452,74 рублей.

В процессе разработки мобильного приложения применены самые современные стандарты написания мобильных приложений в визуальном и функциональном плане. Мобильное приложение упрощает поиск, просмотр и покупку книжной продукции. Возникающие в процессе разработки мобильного приложения проблемы впоследствии решались либо путем более глубокого изучения, либо путем поиска новых путей решения.

Установка мобильного приложения не занимает много времени, так как все данные хранятся на сервере.

Поставленная задача выполнена, все разделы приложения созданы и функционируют. Приложение готово к практическому использованию и может быть дополнено и модернизировано.

Список информационных источников

1. Багласова, Т.Г. Методические указания по выполнению дипломного проекта для учащихся по специальности 2-40 01 01 «Программное обеспечение технологий» / Т.Г. Багласова. – Минск : КБП, 2023. – 30 c.
2. Орлов, С. А. Технологии разработки программного обеспечения: Учебник для вузов. 4-е изд. / С. А. Орлов, Б. Я. Цилькер. – СПб. : Питер, 2012. – 608 с.
3. Общие требования к тестовым документам : ГОСТ 2.105-95. – Введ. 01.01.1996. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 1995. – 84 с.
4. Текст программы. Требования к содержанию, оформлению и контролю качества : ГОСТ 19.401-2000. – Введ. 01.09.2001. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 2000. – 16 с.
5. Документация по Xamarin [Электронный ресурс] / Начало работы с Xamarin.Forms. – Microsoft, 2023. – Режим доступа : https://lern.microsoft.com/ru-ru/xamarin/. – Дата доступа : 30.04.2023.
6. Cреда разработки Visual Studio [Электронный ресурс] / Интегрированая среда разработки – Microsoft, 2022. – Режим доступа : https://docs.microsoft.com/ru-ru/visualstudio/get-started/visual-studio-ide?view=vs-2019. – Дата доступа : 03.05.2023.
7. Microsoft Office [Электронный ресурс] / Microsoft Word: правка документов и общий доступ. – Google, 2020. – Режим доступа : https://play.google.com/store/apps/details?id=com. microsoft.office.word&hl=ru. – Дата доступа : 29.04.2023.
8. Руководство по программированию на C# [Электронный ресурс]. – Microsoft, 2023. – Режим доступа : https://lern.microsoft.com/ru–ru/dotnet/csharp/programming–guide/. – Дата доступа : 02.05.2023.
9. Михнюк, Т.Ф. Охрана труда / Т.Ф. Михнюк. – Минск : ИВЦ Минфина, 2009. – 365 с.
10. Бондарь, А.Г. SQL Server 2012 / А.Г. Бондарь. – СПб. : БХВ-Петербург, 2013. – 608 с.
11. Тепляков, С. Паттерны проектирования на платформе .NET / С.Тепляков. – СПб. : Питер, 2015. – 320 с.

ПРИЛОЖЕНИЕ А  
(обязательное)  
Текст программных модулей

namespace BookShop.mvvm.Model {

public class Book {

//Описание класса Книги

public int Id { get; set; }

public string Name { get; set; }

public float Price { get; set; }

public string Description { get; set; }

public string Genre { get; set; }

public string Author { get; set; }

public string Image { get; set; }

public int CountBooks { get; set; }

public ICommand RemoveBookCommand => new Command(RemoveBookFromBooks);

//функция для удаления книги из корзины

void RemoveBookFromBooks() {

MakeFreeBookFromBasket(this.Id);

//App.ShoppingCartViewModel.Books.Remove(this);

App.ShoppingCartViewModel.Price = App.ShoppingCartViewModel.Books.Sum(x => x.Book.Price \* x.Count);

}

//функция поиска книги по её индексу в базе данных с возвращением Bool значения и объекта класса Book

public static bool FindBook(int idofbook, out Book book) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Книга WHERE Книга.Номер='{idofbook}'", con);

MySqlDataReader result = command.ExecuteReader();

var temp = new Book();

bool flag = false;

while (result.Read()) {

//чтение результата выполнения запроса

temp = new Book {

Id = result.GetInt32("Номер"),

Name = result.GetString("Название"),

Price = result.GetFloat("Цена"),

Description = result.GetString("Описание"),

Genre = result.GetString("Жанр"),

Author = result.GetString("Автор"),

Image = result.GetString("Фото"),

CountBooks = result.GetInt32("Количество")

};

flag = true;

}

con.Close();

book = temp;

return flag;

}

catch (Exception ex) {

Debug.WriteLine(ex.Message);

book = null;

return false;

}

}

//поиск книги по её номеру с возвращением объекта класса Book

public static Book FindBookReturnClassBook(int idofbook) {

var bk = new Book();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Книга WHERE Книга.Номер='{idofbook}'", con);

MySqlDataReader result = command.ExecuteReader();

var temp = new Book();

while (result.Read()) {

//чтение результата выполнения запроса

bk = new Book {

Id = result.GetInt32("Номер"),

Name = result.GetString("Название"),

Price = result.GetFloat("Цена"),

Description = result.GetString("Описание"),

Genre = result.GetString("Жанр"),

Author = result.GetString("Автор"),

Image = result.GetString("Фото"),

CountBooks = result.GetInt32("Количество")

};

}

con.Close();

return bk;

}

catch (Exception ex) {

Debug.WriteLine(ex.Message);

return bk;

}

}

//поиск всех доступных книг для каталога

public static bool FindAllAvailableBooks(out List<Book> books) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Книга WHERE Количество>0", con);

MySqlDataReader result = command.ExecuteReader();

var temp = new List<Book>();

bool flag = false;

while (result.Read()) {

var book = new Book {

Id = result.GetInt32("Номер"),

Name = result.GetString("Название"),

Price = result.GetFloat("Цена"),

Description = result.GetString("Описание"),

Genre = result.GetString("Жанр"),

Author = result.GetString("Автор"),

Image = result.GetString("Фото"),

CountBooks = result.GetInt32("Количество")

};

temp.Add(book);

flag = true;

}

con.Close();

books = temp;

return flag;

}

catch (Exception ex) {

Debug.WriteLine(ex.Message);

books = null;

return false;

}

}

//функция, которая ищет все книги из базы данных и возвращает список объектов

public static ObservableCollection<Book> FindAllBooks() {

ObservableCollection<Book> books = new ObservableCollection<Book>();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Книга", con);

MySqlDataReader result = command.ExecuteReader(); //получение результата от выполнения запроса

while (result.Read()) {

var book = new Book {

Id = result.GetInt32("Номер"),

Name = result.GetString("Название"),

Price = result.GetFloat("Цена"),

Description = result.GetString("Описание"),

Genre = result.GetString("Жанр"),

Author = result.GetString("Автор"),

Image = result.GetString("Фото"),

CountBooks = result.GetInt32("Количество")

};

books.Add(book);

}

con.Close();

return books;

}

catch (Exception ex) {

Debug.WriteLine(ex.Message);

return books;

}

}

//функция, которая ищет все книги из базы данных и возвращает список объектов

public static List<Book> FindAllBooksReport() {

List<Book> books = new List<Book>();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Книга", con);

MySqlDataReader result = command.ExecuteReader(); //чтение результата запроса в базу данных

while (result.Read()) {

var book = new Book {

Id = result.GetInt32("Номер"),

Name = result.GetString("Название"),

Price = result.GetFloat("Цена"),

Description = result.GetString("Описание"),

Genre = result.GetString("Жанр"),

Author = result.GetString("Автор"),

Image = result.GetString("Фото"),

CountBooks = result.GetInt32("Количество")

};

books.Add(book);

}

con.Close();

return books;

}

catch (Exception ex) {

Debug.WriteLine(ex.Message);

return books;

}

}

//функция для проверки необходимого количества книг

public static bool CheckBookReadyForOrder(int id, int count) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT count(\*) FROM Книга WHERE Книга.Номер='{id}' and Книга.Количество>{count}", con);

int res = Convert.ToInt32(command.ExecuteScalar());

con.Close();

if (res != 0)

return true;

return false;

}

catch (Exception e) {

Debug.WriteLine(e.Message);

return false;

}

}

//функция для обновления количества книг в базе данных

public bool UpdateCountBooks() {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"UPDATE Книга SET Количество={CountBooks} WHERE Номер={Id}", con);

command.ExecuteNonQuery();

con.Close();

return true;

}

catch (Exception e) {

Debug.WriteLine(e.Message);

return false;

}

}

//функция для добавления новой кгиги в базу данных

public static bool InsertNewBook(string name, float price, string description, string jenre, string author, string photo, int count) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"INSERT INTO Книга VALUES (0,'{name}',{price},'{description}','{jenre}','{author}','http://185.87.50.136/bookshopimages/{photo}',{count})", con);

command.ExecuteNonQuery();

con.Close();

return true;

}

catch (Exception e) {

Debug.WriteLine(e.Message);

return false;

}

}

//функция, которая возвращает общее количество оставшихся книг

public static int RemainingBooksCount() {

int count = 0;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT SUM(Количество) FROM `Книга`", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

count = result.GetInt32(0); }

con.Close();

return count; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return count; }}}}

namespace BookShop.mvvm.Model {

public class BookOrder {

// Описание класса BookOrder

public Book Book { get; set; }

public int Count { get; set; }

//добавление книг в таблицу, где хранятся данные о книгах к определенному заказу

public static bool MakeBusyBooksByOrder(ObservableCollection<BookOrder> bks, int orderid) {

bool flag = false;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

foreach(var book in bks) {

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"INSERT INTO ЗаказКнига VALUES ({orderid},{book.Book.Id},{book.Count},0)", con);

// запрос в бд

command.ExecuteNonQuery();

con.Close();

MySqlConnection con1 = new MySqlConnection(connStr);

con1.Open();

MySqlCommand command1 = new MySqlCommand($"UPDATE Книга SET Количество=Количество-{book.Count} WHERE Книга.Номер={book.Book.Id}", con1);

command1.ExecuteNonQuery();

con1.Close();

}

return true;

}

catch (Exception e) {

Debug.WriteLine(e.Message);

return flag;

}

}

//функция для поиска книг для определенного заказа

public static ObservableCollection<BookOrder> FindBooksInOrder(int idoforder) {

ObservableCollection<BookOrder> bks = new ObservableCollection<BookOrder>();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT `Книга`.\*, `ЗаказКнига`.`Количество` as Колво FROM `ЗаказКнига` JOIN `Книга` on `Книга`.`Номер`=`ЗаказКнига`.`НомерКниги` WHERE `ЗаказКнига`.`НомерЗаказа`='{idoforder}'", con);

MySqlDataReader result = command.ExecuteReader(); //Получение результата запроса

while (result.Read()) {

bks.Add( new BookOrder {

Book = new Book {

Id = result.GetInt32("Номер"),

Name = result.GetString("Название"),

Price = result.GetFloat("Цена"),

Description = result.GetString("Описание"),

Genre = result.GetString("Жанр"),

Author = result.GetString("Автор"),

Image = result.GetString("Фото"),

},

Count = result.GetInt32("Колво")

});

}

con.Close();

return bks;

}

catch (Exception ex) {

Debug.WriteLine(ex.Message);

return bks;

}

}

//функция для получения количества книг в активных заказах

public static int SoldBooksCount() {

int count = 0;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"select SUM(`ЗаказКнига`.`Количество`) FROM `ЗаказКнига` JOIN `Заказ` ON `Заказ`.`Номер`=`ЗаказКнига`.`НомерЗаказа` WHERE `Заказ`.`Статус`!='Отменён'", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

count = result.GetInt32(0);

}

con.Close();

return count;

}

catch (Exception e) {

Debug.WriteLine(e.Message);

return count;

}

}

//функция для получения количества проданных книг

public static int FindSumCountBooksSelled(string str) {

var price = 0;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT sum(Количество) FROM `ЗаказКнига` WHERE `ЗаказКнига`.`НомерКниги` in ({str})", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

price = result.GetInt32(0);

}

con.Close();

return price;

}

catch (Exception e) {

Debug.WriteLine(e.Message);

return price;

}

}

//команда для удаления книги

public ICommand RemoveBookCommand => new Command(RemoveBookFromBooks);

void RemoveBookFromBooks() {

App.ShoppingCartViewModel.Books.Remove(this);

App.ShoppingCartViewModel.Price = App.ShoppingCartViewModel.Books.Sum(x => x.Book.Price \* x.Count);

}

//команда для изменения количества книги

public ICommand ChangeCountCommand => new Command(ChangeCount);

void ChangeCount(object obj) {

string type = obj as string;

if (type == "Minus") {

if (App.ShoppingCartViewModel.Books[App.ShoppingCartViewModel.Books.IndexOf(this)].Count == 1) {

App.ShoppingCartViewModel.Books.Remove(this);

}

else {

App.ShoppingCartViewModel.Books[App.ShoppingCartViewModel.Books.IndexOf(this)].Count--;

}

}

else {

App.ShoppingCartViewModel.Books[App.ShoppingCartViewModel.Books.IndexOf(this)].Count++;

}

List<BookOrder> bksnew = new List<BookOrder>(App.ShoppingCartViewModel.Books);

App.ShoppingCartViewModel.Books = new System.Collections.ObjectModel.ObservableCollection<BookOrder>(bksnew);

}

}

}

namespace BookShop.mvvm.Model

{

public class Feedback

{

//Описание Класса Feedback

public int ID { get; set; }

public User User { get; set; }

public Book BookFb { get; set; }

public string FeedBack { get; set; }

//Команда чтобы удалить отзыв

public ICommand RemoveFeedBackCommand => new Command(RemoveFeedback);

void RemoveFeedback() {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"DELETE FROM Отзыв WHERE Отзыв.Номер={this.ID}", con);

command.ExecuteNonQuery();

con.Close();

App.AdminFeedbacksViewModel.Feedbacks.Remove(this);

}

catch (Exception e) {

Debug.WriteLine(e.Message);

}

}

//функция для нахождения всех отзывов

public static ObservableCollection<Feedback> FindAllFeedbackForBook(int idbook) {

ObservableCollection<Feedback> feedbacks = new ObservableCollection<Feedback>();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Отзыв WHERE Отзыв.НомерКниги={idbook}", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

feedbacks.Add(new Feedback {

User = User.FindAllUserInfoById(result.GetInt32("НомерПользователя")),

FeedBack = result.GetString("Отзыв")

});

}

con.Close();

return feedbacks;

}

catch (Exception e) {

Debug.WriteLine(e.Message);

return feedbacks;

}

}

//Команда чтобы найти все отзывы

public static ObservableCollection<Feedback> FindAllFeedbacks() {

ObservableCollection<Feedback> feedbacks = new ObservableCollection<Feedback>();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Отзыв ORDER BY Номер DESC", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

feedbacks.Add(new Feedback {

ID = result.GetInt32("Номер"),

User = User.FindAllUserInfoById(result.GetInt32("НомерПользователя")),

BookFb = Book.FindBookReturnClassBook(result.GetInt32("НомерКниги")),

FeedBack = result.GetString("Отзыв")

});

}

con.Close();

return feedbacks;

}

catch (Exception e) {

Debug.WriteLine(e.Message);

return feedbacks;

}

}

//функция для добавления отзыва

public static bool SendFeedback(string feedback, int idbook, int iduser) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"INSERT INTO Отзыв VALUES (0,{iduser},{idbook},'{feedback}')", con);

command.ExecuteNonQuery();

con.Close();

return true;

}

catch (Exception e) {

Debug.WriteLine(e.Message);

return false; }}}

using System;

using System.IO;

using System.Threading.Tasks;

public interface ISave

{

//Интерфейс для переопределения функции сохранения отчетов

Task SaveAndView(DateTime datefrom, DateTime dateto, string parameter);

}

namespace BookShop.mvvm.Model {

public class Order {

//Описание класса Order

public int Id { get; set; }

public int IdCustomer { get; set; }

public float Price { get; set; }

public ObservableCollection<Book> Books { get; set; }

public Dictionary<Book, int> BooksInOrder { get; set; }

public ObservableCollection<BookOrder> BooksOrder { get; set; }

public DateTime DateOfOrder { get; set; }

public DateTime OrderUpdateStateDate { get; set; }

public string Status { get; set; }

//Функция для поиска всех заказов для определенного пользователя

public static List<Order> FindAllOrdersForUser(int idcustomer) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Заказ WHERE НомерПользователя='{idcustomer}'", con);

//запрос в базу данных

MySqlDataReader result = command.ExecuteReader();

var temp = new List<Order>();

while (result.Read()) {

var listbooks = new ObservableCollection<BookOrder>(BookOrder.FindBooksInOrder(result.GetInt32("Номер")));

temp.Add(new Order {

Id = result.GetInt32("Номер"),

IdCustomer = idcustomer,

Price = result.GetFloat("Цена"),

BooksOrder = listbooks,

DateOfOrder = result.GetDateTime("ДатаЗаказа"),

OrderUpdateStateDate = result.GetDateTime("ДатаИзмененияСтатусаЗаказа"),

Status = result.GetString("Статус") });}

con.Close();

return temp; }

catch (Exception ex) {

Debug.WriteLine(ex.Message);

return null; }}

//функция для нахождения всех заказов

public static List<Order> FindAllOrders() {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Заказ ORDER BY Номер DESC", con);

MySqlDataReader result = command.ExecuteReader();

var temp = new List<Order>();

while (result.Read()) {

//чтение результата запроса

var listbooks = new ObservableCollection<BookOrder>(BookOrder.FindBooksInOrder(result.GetInt32("Номер")));

temp.Add(new Order {

Id = result.GetInt32("Номер"),

IdCustomer = result.GetInt32("НомерПользователя"),

Price = result.GetFloat("Цена"),

BooksOrder = listbooks,

DateOfOrder = result.GetDateTime("ДатаЗаказа"),

OrderUpdateStateDate = result.GetDateTime("ДатаИзмененияСтатусаЗаказа"),

Status = result.GetString("Статус")});}

con.Close();

return temp; }

catch (Exception ex) {

Debug.WriteLine(ex.Message);

return null; }}

//функция для нахождения последнего заказа пользователя

public static Order FindLastUserOrder(int userid) {

var order = new Order();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM `Заказ` WHERE `Заказ`.`НомерПользователя`={userid} ORDER BY `Заказ`.`Номер` DESC LIMIT 1", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

var listbooks = new ObservableCollection<BookOrder>(BookOrder.FindBooksInOrder(result.GetInt32("Номер")));

order = new Order {

Id = result.GetInt32("Номер"),

IdCustomer = result.GetInt32("НомерПользователя"),

Price = result.GetFloat("Цена"),

BooksOrder = listbooks,

DateOfOrder = result.GetDateTime("ДатаЗаказа"),

OrderUpdateStateDate = result.GetDateTime("ДатаИзмененияСтатусаЗаказа"),

Status = result.GetString("Статус")};}

con.Close();

return order; }

catch (Exception ex) {

Debug.WriteLine(ex.Message);

return order; }}

//функция создания заказа

public static int MakeOrder(int idcustomer, float price) {

int res = -1;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"INSERT INTO Заказ VALUES (0,'{idcustomer}','{price}','{DateTime.Now.ToString("yyyy-MM-dd H:mm:ss")}','{DateTime.Now.ToString("yyyy-MM-dd H:mm:ss")}','Оформлен');", con);

command.ExecuteNonQuery();

con.Close();

MySqlConnection con1 = new MySqlConnection(connStr);

con.Open();

MySqlCommand command1 = new MySqlCommand($"SELECT Номер FROM Заказ WHERE НомерПользователя='{idcustomer}' ORDER BY Номер DESC LIMIT 1", con);

//запрос в базу данных

var findid = command1.ExecuteReader();

while (findid.Read())

res = findid.GetInt32("Номер");

return res; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return res; }}

//функция обновления статуса заказа

public static bool UpdateOrderStatus(int id, string status) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"UPDATE Заказ SET Статус='{status}', ДатаИзмененияСтатусаЗаказа='{DateTime.Now.ToString("yyyy-MM-dd H:mm:ss")}' WHERE Номер={id}", con);

command.ExecuteNonQuery();

con.Close();

return true; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return false; }}

//функция для получения количества всех заказов

public static int AllOrdersCount() {

int count = 0;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"select count(\*) from Заказ", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

count = result.GetInt32(0); }

con.Close();

return count; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return count; }}

//функция для получения количества отклоненных заказов

public static int RejectedOrdersCount() {

int count = 0;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"select count(\*) from Заказ WHERE Статус='Отменён'", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

count = result.GetInt32(0); }

con.Close();

return count; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return count; }}

//функция для получения количества активных заказов

public static int ActiveOrdersCount() {

int count = 0;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"select count(\*) from Заказ WHERE Статус!='Отменён' and Статус!='Оформлен' and Статус!='Завершён'", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

count = result.GetInt32(0); }

con.Close();

return count; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return count; }}

//функция для получения номеров заказов в определённый промежуток времени

public static List<int> FindIdsOfOrdersInDateTime(DateTime df, DateTime dt) {

var listofids = new List<int>();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT `Заказ`.`Номер` FROM `Заказ` WHERE `Заказ`.`ДатаЗаказа`>='{df.ToString("yyyy-MM-dd")}' and `Заказ`.`ДатаИзмененияСтатусаЗаказа`<='{dt.ToString("yyyy-MM-dd")}' and `Заказ`.`Статус`='Завершён'", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

listofids.Add(result.GetInt32(0)); }

con.Close();

return listofids; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return listofids; }}

//функция для получения суммы для заказов

public static double FindSumPriceForOrders(string str) {

double price = 0;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT SUM(Цена) FROM `Заказ` WHERE `Заказ`.`Номер` in ({str})", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

price = result.GetDouble(0); }

con.Close();

return price; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return price; }}}

namespace BookShop.mvvm.Model

{

//Описание класса Support

class Support{

public int Id { get; set; }

public int IdUser { get; set; }

public string Question { get; set; }

public string Answer { get; set; }

//функция для создания запроса в поддержку

public static bool CreateRequest(int userid, string question) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"INSERT INTO Поддержка VALUES(0,{userid},'{question}','-')", con);

command.ExecuteNonQuery();

con.Close();

return true; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return false; }}

//функция для получения запросов в поддержку определенного пользователя

public static ObservableCollection<Support> FindUserRequests(int userid) {

ObservableCollection<Support> books = new ObservableCollection<Support>();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Поддержка WHERE НомерПользователя={userid}", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

var book = new Support {

Id = result.GetInt32("Номер"),

IdUser = result.GetInt32("НомерПользователя"),

Question = result.GetString("Вопрос"),

Answer = result.GetString("Ответ")};

books.Add(book); }

con.Close();

return books; }

catch (Exception ex) {

Debug.WriteLine(ex.Message);

return books; }}

//Функция для получения не отвеченных запросов в поддержку

public static ObservableCollection<Support> FindAdminRequests() {

ObservableCollection<Support> books = new ObservableCollection<Support>();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM Поддержка WHERE Ответ='-'", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

var book = new Support {

Id = result.GetInt32("Номер"),

IdUser = result.GetInt32("НомерПользователя"),

Question = result.GetString("Вопрос"),

Answer = result.GetString("Ответ")};

books.Add(book); }

con.Close();

return books; }

catch (Exception ex) {

Debug.WriteLine(ex.Message);

return books; }}

//функция для ответа на вопрос в поддержке

public static bool UpdateRequestAnswer(int idquestion,string answer) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"UPDATE Поддержка SET Ответ = '{answer}' WHERE Номер ={idquestion}", con);

command.ExecuteNonQuery();

con.Close();

return true; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return false; }}}}

namespace BookShop.mvvm.Model {

public class User {

//Описание класса User

public int Id { get; set; }

public string Name { get; set; }

public string Surname { get; set; }

public string Email { get; set; }

public string Password { get; set; }

public string Role { get; set; }

public DateTime RegistrationDate { get; set; }

//Свойсто для формирования строки из фамилии и имени

public string Fullname {

get {return Surname + " " + Name; }}

//функция для проверки на пользователя с определенной почтой

public static bool CheckForNotExistingEmail(string checkemail) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT count(\*) FROM Пользователь WHERE Пользователь.Почта='{checkemail}'", con);

int res = Convert.ToInt32(command.ExecuteScalar());

con.Close();

if (res != 0)

return false;

return true;

}catch(Exception e) {

Debug.WriteLine(e.Message);

return false; }}

//Создание нового пользователя

public static bool CreateNewUser(string nameuser, string surnameuser, string emailuser, string passworduser) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"INSERT INTO Пользователь VALUES(0,'{nameuser}','{surnameuser}','Покупатель','{passworduser}','{emailuser}','{DateTime.Now.ToString("yyyy-MM-dd H:mm:ss")}')", con);

command.ExecuteNonQuery();

con.Close();

return true; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return false; }}

//Получение всей информации о пользователе

public static bool FindAllUserInfo(string emailuser, string passworduser, out User createdUser) {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM `Пользователь` WHERE `Пользователь`.`Почта`='{emailuser}' and `Пользователь`.`Пароль`='{passworduser}'", con);

MySqlDataReader result = command.ExecuteReader();

var temp = new User();

bool flag = false;

while (result.Read()) {

temp = new User {

Id = result.GetInt32("Номер"),

Name = result.GetString("Имя"),

Surname = result.GetString("Фамилия"),

Email = result.GetString("Почта"),

Password = result.GetString("Пароль"),

Role = result.GetString("Роль"),

RegistrationDate = result.GetDateTime("ДатаРегистрации")

};

flag = true; }

con.Close();

createdUser = temp;

return flag; }

catch (Exception e) {

Debug.WriteLine(e.Message);

createdUser = new User();

return false; }}

//Поиск информации о пользователе по его номеру

public static User FindAllUserInfoById(int id) {

var user = new User();

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT \* FROM `Пользователь` WHERE Номер={id}", con);

//запрос в базу данных

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

user = new User {

Id = result.GetInt32("Номер"),

Name = result.GetString("Имя"),

Surname = result.GetString("Фамилия"),

Email = result.GetString("Почта"),

Password = result.GetString("Пароль"),

Role = result.GetString("Роль"),

RegistrationDate = result.GetDateTime("ДатаРегистрации")};}

con.Close();

return user; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return user; }}

//Функция для получения количества пользователей

public static int UsersCount() {

int count = 0;

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"SELECT COUNT(\*) FROM `Пользователь`", con);

MySqlDataReader result = command.ExecuteReader();

while (result.Read()) {

count = result.GetInt32(0); }

con.Close();

return count; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return count; } }

//обновление информации пользователя

public bool UpdateInfoUser() {

try {

string connStr = "server=185.87.50.136;user=\*\*\*\*\*\*\*\*\*\*;database=КнижныйМагазин;password=\*\*\*\*\*\*\*\*\*\*;";

MySqlConnection con = new MySqlConnection(connStr);

con.Open();

MySqlCommand command = new MySqlCommand($"UPDATE Пользователь SET Имя='{Name}', Фамилия='{Surname}', Пароль='{Password}' WHERE Номер={Id}", con);

command.ExecuteNonQuery();

con.Close();

return true; }

catch (Exception e) {

Debug.WriteLine(e.Message);

return false; }}}}

ПРИЛОЖЕНИЕ Б  
(справочное)  
Результат работы приложения

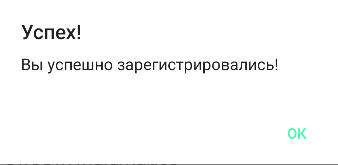


Рисунок Б.1 – Результат успешной регистрации

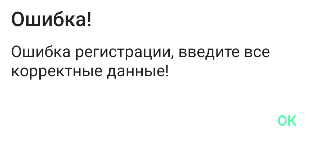


Рисунок Б.2 – Результат неправильного ввода данных в форме регистрации

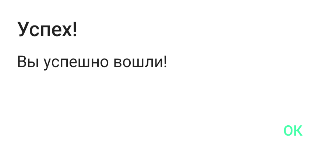


Рисунок Б.3 – Результат успешной авторизации

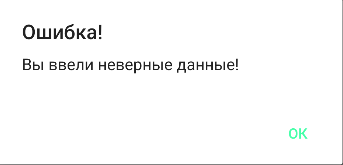


Рисунок Б.4 – Результат неправильного ввода данных в форме авторизации

ДП Т.992136.401 ГЧ

Т. Контр.

Н.Контр.

Утверд.

КБиП

Листов 3

Провер.

Разраб.

Подп. и дата

Инв.№дубл.

№ докум.

Подпись

Дата

Изм.

Лист

Взам.инв.№

Подп. и дата

РЕАЛИЗАЦИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ УЧЕТА ПРОДАЖ КНИЖНОЙ ПРОДУКЦИИ ООО «АРБИС-БЕЛ»

*Структура мобильного приложения*

У

Масса

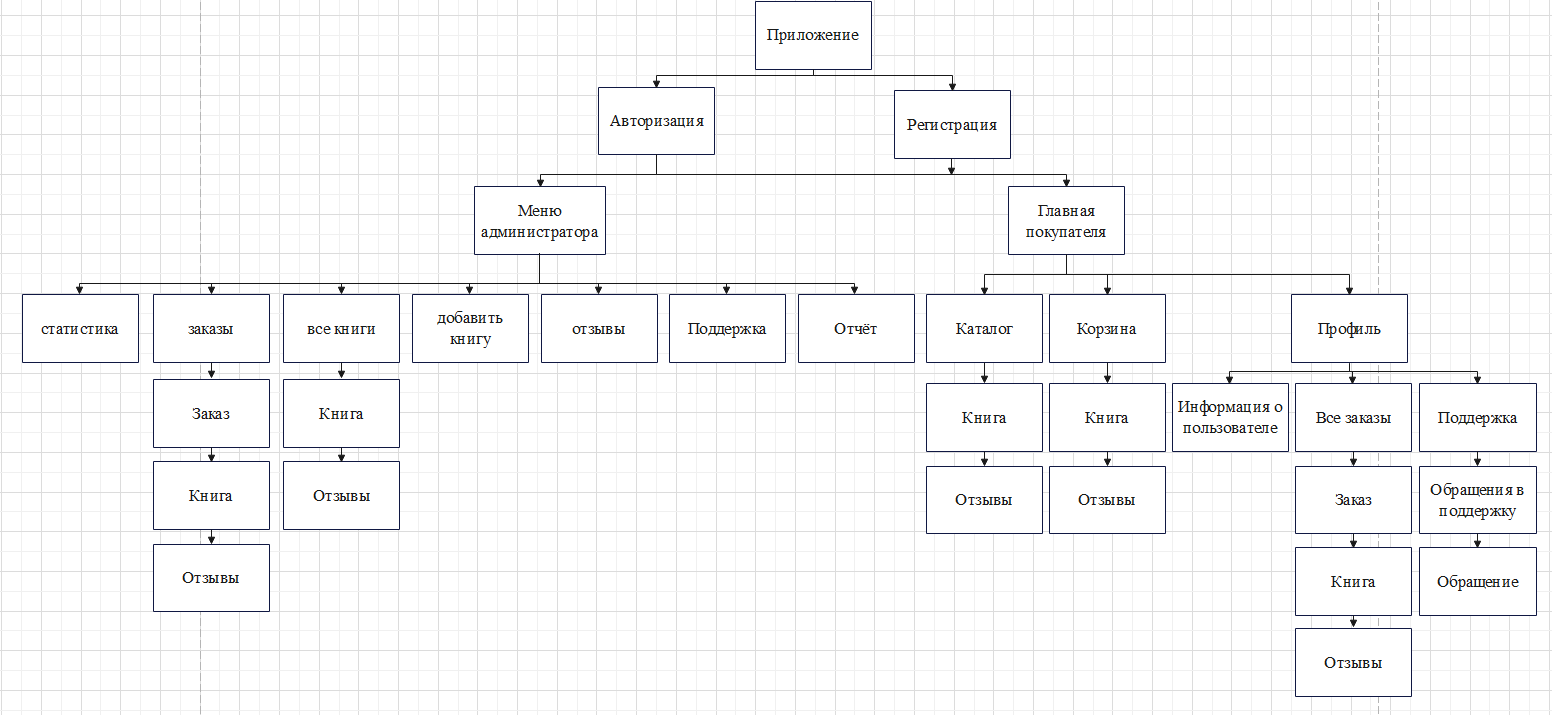
Лит.

Масштаб

Лист 1

Реценз.





*Багласова Т.Г.*

*Басалыга Л.В.*

*Чужавко Е.А.*

*Басалыга Л.В.*

*Багласова Т.Г.*

ДП Т.992136.401 ГЧ

Т. Контр.

Н.Контр.

Утверд.

КБиП

Листов 3

Провер.

Разраб.

Подп. и дата

Инв.№дубл.

№ докум.

Подпись

Дата

Изм.

Лист

Взам.инв.№

Подп. и дата

РЕАЛИЗАЦИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ УЧЕТА ПРОДАЖ КНИЖНОЙ ПРОДУКЦИИ ООО «АРБИС-БЕЛ»

*Диаграмма вариантов использования*

У

Масса

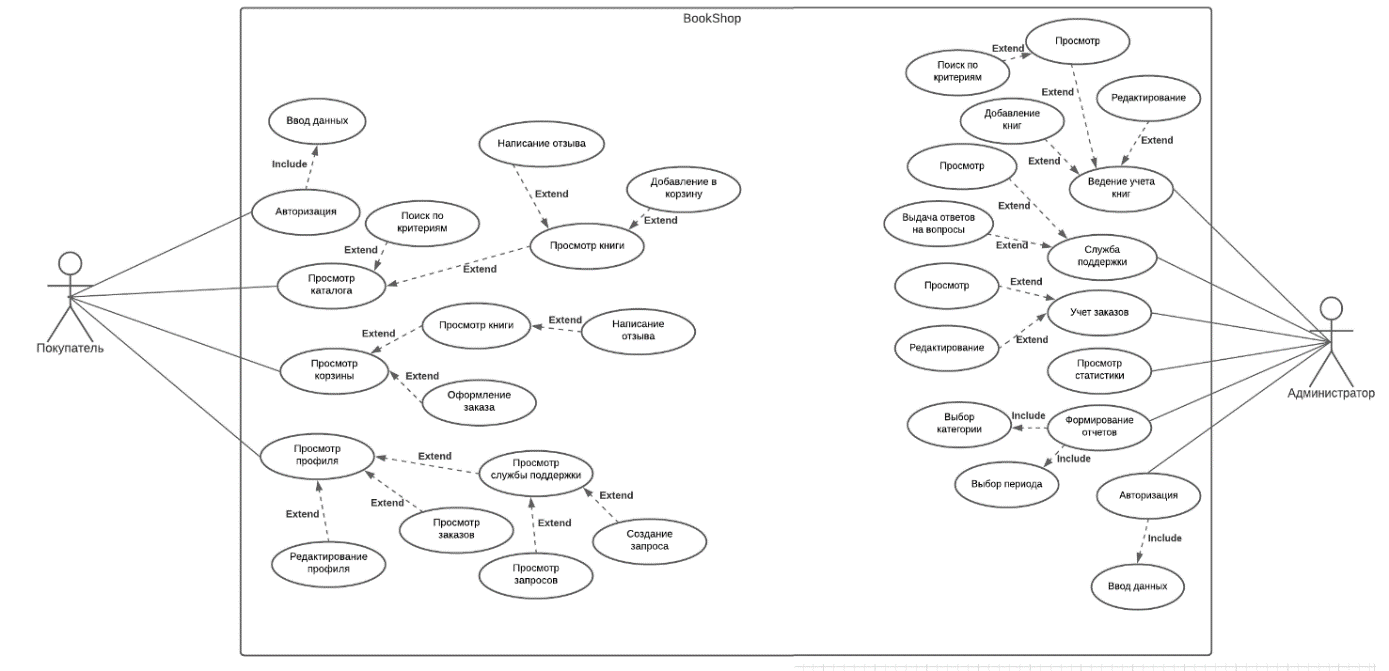
Лит.

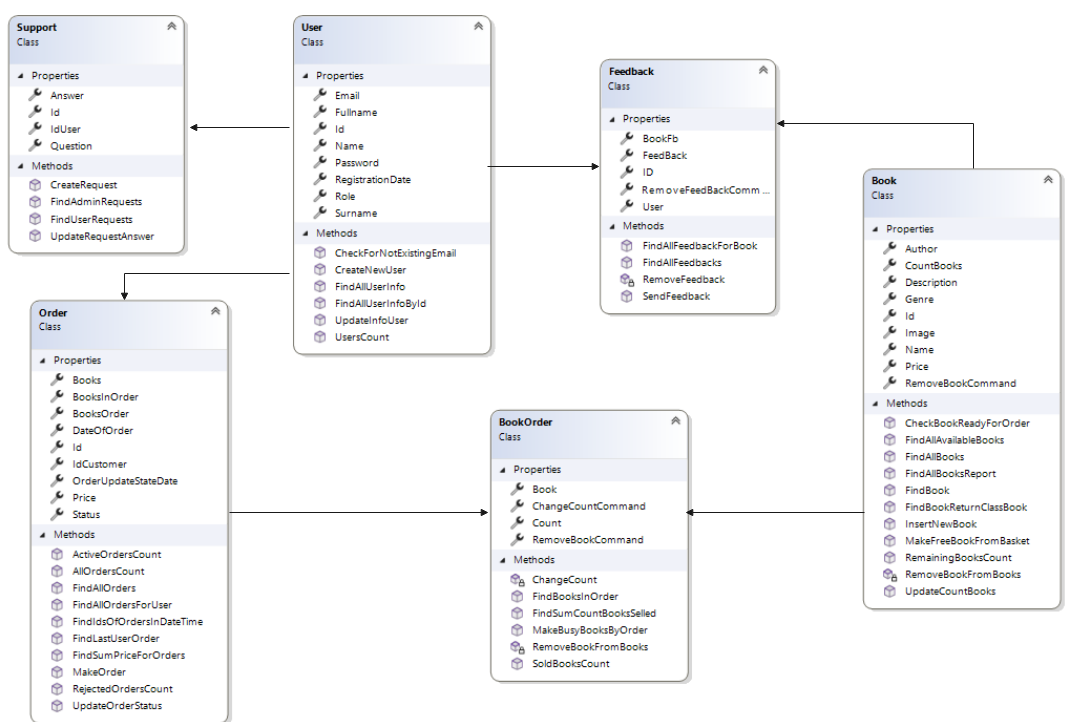
Масштаб

Лист 2

Реценз.







*Багласова Т.Г.*

*Багласова Т.Г.*

*Чужавко Е.А.*

*Басалыга Л.В.*

*Басалыга Л.В.*

ДП Т.992136.401 ГЧ

Т. Контр.

Н.Контр.

Утверд.

КБиП

Листов 3

Провер.

Разраб.

Подп. и дата

Инв.№дубл.

№ докум.

Подпись

Дата

Изм.

Лист

Взам.инв.№

Подп. и дата

РЕАЛИЗАЦИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ УЧЕТА ПРОДАЖ КНИЖНОЙ ПРОДУКЦИИ ООО «АРБИС-БЕЛ»

*Диаграмма классов*

У

Масса

Лит.

Масштаб

Лист 3

Реценз.



*Чужавко Е.А.*

*Багласова Т.Г.*

*Багласова Т.Г.*

*Басалыга Л.В.*

*Басалыга Л.В.*