Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Филиал

«Минский радиотехнический колледж»

Допущен к защите

Заведующий отделением

компьютерных технологий

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( С.И. Иванова )

«\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_20 г.

Социальная сеть с возможностью передачи файлов "TalkIn"

Пояснительная записка

ДП62492.021130.081ПЗ

Дипломник /Н.В. Ширяев /

Руководитель /А.В. Гордеюк/

Консультант по экономической части /А.А. Викторова /

Консультант по охране труда /Т.В. Горошко /

Нормоконтроль /А.В. Гордеюк /

Председатель ЦК / И.Г. Смолер /

Рецензент / /

2020

Иизм

Министерство образования Республики Беларусь

Содержание

Изм.

Лист



№ докум.

Подп.

Дата

Лист

2

ДП42492.024102.081ПЗ

Разраб.

Ширяев Н.В.

Пров.

Карпович Д.В.

Реценз.

Н. Контр.

Гордеюк А.В.

Утв.

Смолер И.Г.

Социальная сеть с возможностью передачи файлов "TalkIn"

Пояснительная записка

Лит.

Листов

111

МРК

[Введение 3](#_Toc327105829)

[1 Постановка задачи](#_Toc327105830) 6

[1.1 Описание предметной области 6](#_Toc327105831)

[1.2 Обзор существующих аналогов 8](#_Toc327105832)

[1.3 Информационная база задачи 9](#_Toc327105833)

[1.4 Функциональное назначение 10](#_Toc327105833)

[2 Проектирование задачи 12](#_Toc327105834)

[2.1 Алгоритм решения задачи 12](#_Toc327105835)

[2.2 Логическое моделирование 13](#_Toc327105836)

[2.3 Выбор и обоснование инструментов разработки 17](#_Toc327105837)

[3 Программная реализация 21](#_Toc327105838)

[3.1 Физическая структура 21](#_Toc327105839)

[3.2 Описание разработанных модулей 25](#_Toc327105842)

[4 Тестирование 28](#_Toc327105846)

[5 Применение 35](#_Toc327105849)

[5.1 Назначение и условия применения 35](#_Toc327105850)

[5.2 Руководство пользователя 36](#_Toc327105851)

[6 Охрана труда 42](#_Toc327105853)

[7 Экономический раздел 47](#_Toc327105877)

[Заключение 66](#_Toc327105878)

[Список использованных источников 67](#_Toc327105879)

[Приложение А Текст программы 68](#_Toc327105880)

# Введение

Пожалуй, всем известны имена таких web-сайтов, как Одноклассники.ру и ВКонтакте.ру. Эти ресурсы, называемые «социальными сетями», являются самыми посещаемыми сайтами в российском сегменте сети Интернет. Их аудитория составляет десятки миллионов человек, трафик исчисляется терабайтами, а количество серверов – сотнями. В мировой сети тоже имеются свои гранды, такие как MySpace.com и Facebook.com, которые посещают более 200 миллионов человек в месяц [1].

Эти ресурсы выполняют функцию социального взаимодействия людей в сети Интернет. Среди их возможностей можно выделить ведение собственного профиля (информация о пользователе, местах его учебы, работы, и. т. д.), добавление других пользователей в список контактов, добавление фотографий в личный фотоальбом и просмотр альбомов других участников, загрузка файлов различных типов, таких как видеофайлы, аудиофайлы, возможность просмотра видео и прослушивания аудиокомпозиций, ведение личного блога, и многие другие функции.

В связи с развитием новой эры в мире Web, носящей название Web 2.0, начало появляться большое количество подобных проектов. Самые крупные из них, такие как уже упоминавшиеся Myspace.com и FaceBook.com, стали не только социальным, но и коммерческим феноменом. Ежегодная выручка компании-владельца FaceBook.com оценивается в 300 миллионов долларов.

Однако, несмотря на популярность существующих социальных сетей, в каждой из них есть свои недостатки. Причем недостатки имеются как в функциональности (недостаточная гибкость настроек пользовательского профиля, малое количество доступных сервисов или их платный статус), так и в удобстве взаимодействия пользователя с системой. Поэтому актуальной является идея создания социальной сети нового поколения.

Целью настоящей работы является создание функциональной и удобной для пользователя социальной сети, которая по своим эксплуатационным качествам смогла бы успешно конкурировать с существующими проектами подобного рода.

Из всего вышеперечисленного следует, что актуальность такого программного продукта высока. Вследствие этого были поставлены следующие задачи:

* разработать простой, но в тоже время многофункциональный интернет-магазин;
* создать приложение с двумя типами доступа - пользователь и администратор;
* разработать приятный и понятный дизайн, для лучшего восприятия клиентом;
* разработать интерактивный сайт.

В данном дипломном проекте поставлена цель создать веб-приложение радиоуправляемых технологий, который удобнее всего реализовать с помощью базы данных, в которой будет храниться информация о товаре, о клиентах и о заказах. Такое хранение данных обусловлено тем, что использование баз данных существенно облегчает работу над данными, а также укоряет ее.

В разделе «Постановка задачи» описана предметная область программного средства, проведен обзор существующих аналогов, сформулированы цели и требования к программному средству, описано функциональное назначение программы и определены входные выходные данные.

Раздел «Проектирование задачи» описывает проектирование интерфейса, проводится логическое представление данных, а также обосновывается выбор инструментов для разработки.

Раздел «Разработка программного средства» описывает структуру программного средства. Приводится описание исходных файлов проекта, а также описываются разработанные классы, включающие в себя модели, представления и контроллеры.

В разделе «Тестирование» рассмотрены виды тестирования и их описание. Описаны функциональные тесты, которые проводились для выявления некорректной работы разрабатываемого программного средства.

В разделе «Назначение и область применения программы» рассмотрена пошаговая инструкция работы с автоматизированной системой продаж строительных материалов.

В «Экономическом разделе» рассчитаны затраты на производство, прибыль, отпускная цена, а также подробно рассчитаны основные экономические величины, показывающие рентабельность приложения.

В разделе «Охрана труда» рассмотрены вопросы трудоохранного анализа информационной технологии и её оздоровление, описываются особенности умственного труда пользователя при использовании созданной технологии. Определены влияния элементов рабочего места в организации труда на работоспособность и здоровье операторов, и описан рациональный режима труда и отдыха для снижения умственного утомления.

Графическая часть пояснительной записки представлена в виде диаграммы вариантов использования и диаграммы деятельности.

# 1 Постановка задачи

Система должна позволять работать как клиентам, так и сотрудникам предприятия, поэтому, необходимо выполнить решение в двух составляющих: режим клиента и режим администратора.

Основным назначением системы является повышение скорости обслуживания клиентов магазина и уменьшение числа ошибок, связанных с оформлением вновь поступившего товара и его продажи, а также получение оперативной информации о состоянии склада.

Необходимо разработать удобное и простое в освоении программное средство для решения поставленных задач. Основной задачей решения должно являться ускорение процессов продажи и контроль правильности. Решение должно представлять собой систему на базе клиент-серверной архитектуры. Клиент-серверная архитектура, будет реализована на базе SQL сервера.

Задачи серверной части:

* организация доступа клиентов к базе данных;
* разграничение прав доступа;
* аутентификация пользователей;
* отправка и запрос данных из БД;
* задачи клиентской части;
* передача и запрос данных с сервера;

Так как решение является многопользовательским, то в системе должна быть учетная запись Администратора. Администратор имеет неограниченные права доступа к данным и всем функциям системы.

# Описание предметной области

Разрабатываемое программное средство предназначено для осуществления переписки между пользователями. Возможность добавления друзей, написание статей, редактирование информации о пользователе, просмотр страниц других пользователей, регистрация новых аккаунтов, (разные формы доступа) программный продукт (ПП) должен обеспечивать безопасное использование продукта. Удобный интерфейс обеспечит комфортное использование программного продукта с минимальными затратами времени на изучение всех функций и возможностей.

Запись – комбинация из картинки, контентной части, названия записи, а также полей автора и время создания, которые автоматически выставляются на уровне серверной части.

Автор – пользователь, который опубликовал запись.

Пользователь социальной сети – это человек, который зарегистрировался и представил свои персональные данные, а также, прошедший идентификацию в одной из социальных сетей, существующих в виртуальном пространстве интернета. Для самого пользователя он является центром сети и всего происходящего, а другие пользователи, его друзья, контент ветками, которые направлены в различных направлениях и позволяют общаться, высказывать свои мнения, а также делиться последней информацией интересующей тематики.

Лайк  – это условное выражение одобрения материалу, пользователю, фотографии, выражающиеся нажатием одной кнопки.

Дизлайк – это выражение антипатии или неодобрения к опубликованному контенту. Дизлайк – это полная противоположность лайку.

1.2 Обзор существующих аналогов

Аналог – это то, что представляет собою подобие чего-либо, соответствие какому-либо предмету, явлению, понятию.

Приведем несколько аналогов данного приложения.

Для обзора были выбраны следующие аналоги: «ВКонтакте», «Facebook», LinkedIn.

«ВКонтакте» российская социальная сеть со штаб-квартирой в Санкт-Петербурге. Сайт доступен на более чем 90 языках; особенно популярен среди русскоязычных пользователей. «ВКонтакте» позволяет пользователям отправлять друг другу сообщения, создавать собственные страницы и сообщества, обмениваться изображениями, тегами, аудио- и видеозаписями, играть в браузерные игры.

Дизайн сайта аналогичен внешнему виду всех социальных сетей, хорошо продуман, наглядно и просто реализован выбор параметровю. Комфортно провести время за общением с друзьями не получится из-за большого количества рекламы. На сайте вам будут доступны следующие функции:

* моя страница;
* новости;
* сообщения;
* друзья;
* группы;
* фотографии;
* музыка;
* игры

На рисунке 1.1 представлена главная страница приложения «Вконтакте»

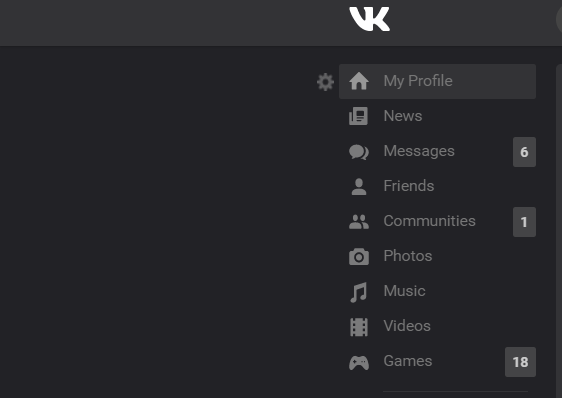


Рисунок.1.1 - главная страница приложения «Вконтакте»

Ещё одной социальной сетью является Facebook. Если «Вконтакте» располагает широкий русскоязычной аудиторией, то «Facebook» почти полностью ориентирован на людей из зарубежных стран. При этом структура сайта имеет следующий вид:

* поиск;
* главная;
* профиль;
* аккаунт;
* публикации по теме;

На рисунке 1.2 представлена главная страница приложения «Все вместе».

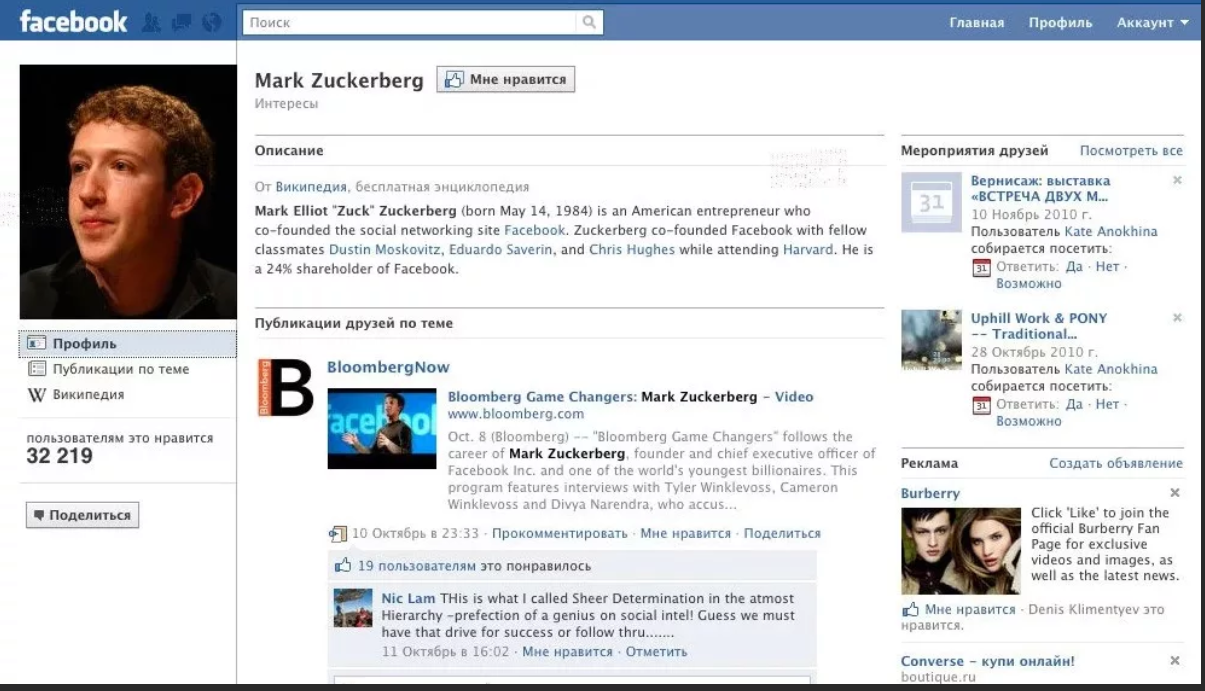


Рисунок.1.2 - главная страница приложения «Facebook»

Но при этом страница загромождена ненужной информацией, реклама прям на главной странице, появляется прям во время переписки или прослушивания музыки, что очень неудобно.

Следующий сайт-аналог ­­- это «LinkedIn». Эта социальная сеть предназначена для общения профессионалов, а не для развлечения. Это и отличает её от других соцсетей, таких как Facebook, Одноклассники, ВКонтакте. Здесь можно наладить деловые связи с коллегами, найти вакансии. В ней HR-менеджерам удобно находить специалистов для привлечения, а пользователям – быть в курсе новостей, касающихся профессиональных вопросов. Бизнес-страницы компаний, со статьями и новостями, тематическое общение, совместное решение сложных вопросов – всё это варианты использования этого портала. Список функций на сайте:

* главная;
* сеть;
* вакансии;
* сообщения;
* уведомления»;
* профиль;
* для работы;
* поиск контактов;

На рисунке 1.3 представлена главная страница приложения «LinkedIn».

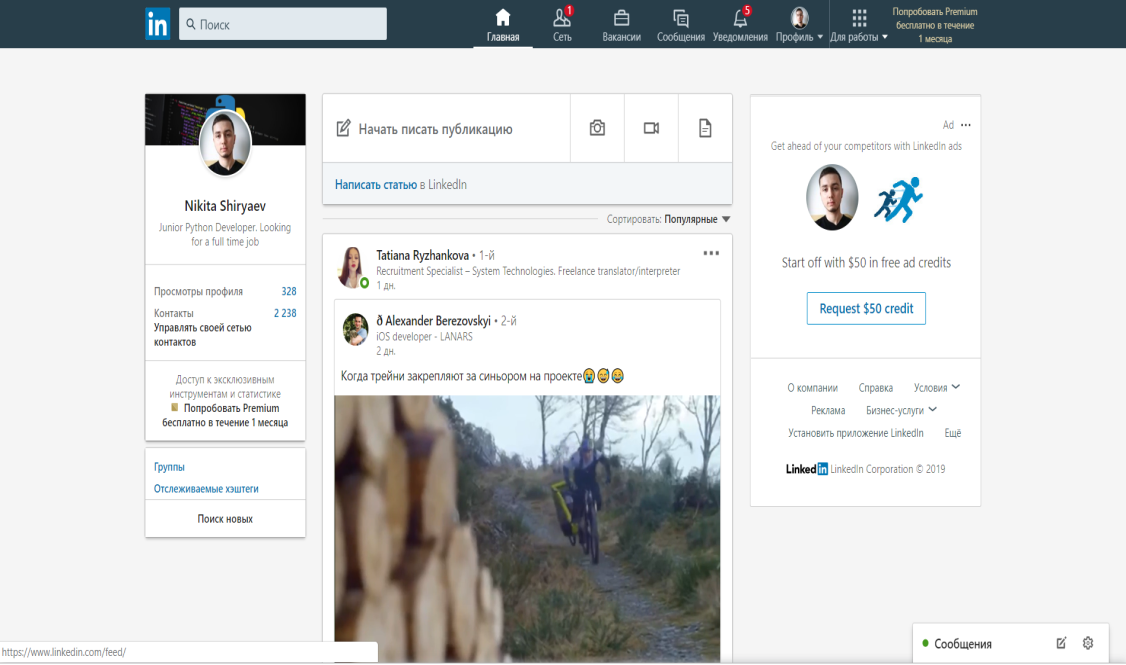


Рисунок.1.2 - главная страница приложения «LinkedIn»

Таким образом, чтобы превзойти аналогичные продукты, и исходя из требований пользователя, основной упор в разработке был сделан на удобство и простоту пользовательского интерфейса, отсутствие рекламы, нет высокой загруженности сети.

1.3 Информационная база задачи

Функцией написания статьи может заниматься любой зарегистрированный пользователь на сайте. Также на сайте можно вести профиль, просматривать чужие запсиси, комментировать и оценивать их. При создании записи требуется перейти на свою, нажать на кнопку «Создать запись» и заполнить нижеперечисленные поля:

* название записи;
* текст записи;
* картинка записи.

Пользователь имеет возможность редактировать и удалять запись, в которой он является автором.

Пользователь также обладает возможностью оценить запись. Для оценки записи ему требуется зайти на страницу с самим этим объектом и под контентной частью нажать на кнопку нравится или не нравится с иконкой «Палец вверх» или «Палец вниз».

Пользователь имеет возможность добавить другого пользователя в друзья. Для этого нужно перейти на страницу пользователя и нажать кнопку «Добавить в друзья»

Также пользователь имеет возможность отправить сообщение другому пользователю, зарегистрированному в системе. Чтобы написать сообщение, нужно перейти на страницу другого пользователя и нажать «send message».

Интерфейс программы должен отвечать следующим требованиям:

* удобный;
* адаптивный;
* фильтрация данных;
* простой в использование.

Дизайн пользовательского интерфейса должен корректно отображаться и работать на компьютерах, ноутбуках, планшетах и телефонах.

Интерфейс системы должен поддерживать русский язык.

Программное средство должно иметь необходимые сообщения при добавлении, удаление или изменение данных. Так же сообщение при авторизации при неверном логине или пароле.

Информационная база задачи состоит из входной, выходной и постоянной информации.

Входная информация — информация, поступающая в систему в виде документов, данных, сообщений, сигналов, необходимая для выполнения функций.

Выходная информация — информация, получаемая в результате выполнения функций системы и выдаваемая на объект ее деятельности, пользователю или в другие системы.

Постоянная информация — информация, которая остается неизменной в течение длительного периода времени и многократно используется при обработке переменной информации.

Входные данные разрабатываемого программного средства:

* информация для входа на личную страницу;
* текст сообщения;
* адрес получателя;
* логин пароль для регистрации;
* БД с информацией о пользователях.

Выходные данные разрабатываемого программного средства:

* информация, отображаемая на странице пользователя;
* измененная база данных пользователей;
* передаваемые сообщения.

Постоянными данными являются:

* форма для добавления поста;
* форма для редактирования поста.

Информационная база - это совокупность упорядоченной информации, которая используется для функционирования системы. Как и любая компьютерная программа, данная программа обрабатывает входные данные и выводит результат в виде выходных данных. Входные данные имеют структуру любой задачи - это описание некоторой ситуации (исходные данные) и формулировка вопроса, ответ на который должен быть на основе сложившейся ситуации.

1.4 Функциональное назначение

Разрабатываемое программное средство предназначено для осуществления переписки между пользователями.

Предпологаемые задачи работы в приложении:

* переписка;
* добавление в друзья;
* просмотр страницы пользователя;
* создание постов;
* редактирование постов;
* регистрация;
* вход на сайт;
* оценка поста.

Разрабатываемые алгоритмы должны легко модифицироваться. В течении сроков, установленных в договоре между Заказчиком и Разработчиком, в ПП могут быть внесены изменения, не требующие глобальной переработки алгоритма.

На этапе общесистемного проектирования одной из главнейших задач является задача проектирования структуры. Применительно к человеко-машинным системам, наиболее распространённым является случай, когда принципы построения системы, выполняемые функции и компоненты системы оказываются заданными.

Эффективным инструментом проектирования и рационализации организационных структур управления является моделирование, позволяющее находить оптимальные варианты их построения, прогнозировать их развитие, проводить оперативную диагностику состояния действующей структуры и устанавливать ее соответствие реальным производственно-технологическим условиям, оценивать различные варианты построения организационной структуры, когда прямые эксперименты невозможны или затруднительны, а также экономически невыгодны, а иногда невозможны.

2 Проектирование задачи

2.1 Алгоритм решения задачи

Для составления алгоритма задачи, необходимо понять логический ход работы программного средства.

При разработке программного продукта были выделены файлы описания стилей, форм:

- base.html – файл, описывающий макет разметки всех страниц;

- style.css – файл, позволяющий регулировать оформление блоков страницы (цвет, размер шрифта, отступы).

Так же были разработаны следующие классы:

- manage.py – файл для запуска приложения;

- LoginRequiredMixin – класс для авторизации пользователя. Содержит следующие методы:

1) LoginRequiredMixin.handle\_no\_permission() – метод для проверки прав пользователя к запрашиваемому объекту;

2) LoginRequiredMixin.get\_login\_url() – метод для получения url на страницу с авторизацией;

- UsersListView – класс для отображения пользователей зарегистрированных в системе;

- PostEditView – класс для редактирования записи от авторизованного пользователя. Содержит следующие методы:

1) PostEditView.get() – метод, который отправляет пользователю форму на редактирование записи;

2) PostEditView.port() – метод для обработки POST запросов, редактирует статью в соответствие с валидными данными, которые пришли на адрес записи;

- RegisterView – класс для создания пользователей от авторизованного пользователя. Содержит следующие методы:

1) RegisterView.get() – метод, который отправляет пользователю форму на регистрацию;

2) RegisterView.post() – метод для обработки POST запросов, создает запись нового пользователь в базе данных.

- Like/Dislike – модель таблицы «Like/Dislike» для оценок. Содержит следующие методы:

1) Like/Dislike.delete\_or\_switch() – метод для удаления и изменения оценки;

2) Like/Dislike.get\_related\_likes() – метод для получения количества всех положительных оценок к оцененному объекту;

3) Like/Dislike. get\_related\_dislikes() – метод для получения количества всех отрицательных оценок к оцененному объекту.

- MessageView() – модель таблицы «Message» хранения сообщений пользователей. Содержит следующие методы:

1) MessageView.post()– метод для обработки POST запросов;

2) MessageView.get()– метод для обработки GET запросов;

2.2 Логическое моделирование

На этапе логического моделирования данных необходимо разработать логическую модель для физической реализации базы данных.

Для этого необходимо решить ряд задач, таких как:

* выбор модели данных;
* нормализация таблиц;
* проверка логической модели данных на предмет возможности выполнения всех транзакций;
* определение требований поддержки целостности данных и их документирование;
* создание окончательного варианта логической модели данных.

Исходя из описания предметной области, организацию данных можно представить набором следующих отношений:

* профиль;
* пользователь;
* запись;
* друзья;
* лайк;
* дислайк;

Далее следует идентифицировать связи.

Связь – это некоторая ассоциация между двумя отношениями. Связи позволяют по одному отношению находить другие отношения, связанные с ним. Они бывают следующих типов:

* один-к-одному;
* один-ко-многим;
* многие-к-одному;
* много-ко-многим.

Связь типа один-к-одному означает, что один экземпляр одного отношения связан с одним экземпляром второго отношения.

Связь типа один-ко-многим означает, что один экземпляр одного отношения связан с несколькими экземплярами второго отношения. Данный вид связи используется для создания логического моделирования автоматизированной системы продаж строительных материалов. Первое отношение (со стороны «один») называется родительским, второе (со стороны «много») — дочерним.

Связь типа многие-к-одному означает, что несколько экземпляров одного отношения связан с одним экземпляром второго отношения.

Связь типа много-ко-многим означает, что каждый экземпляр одного отношения может быть связан с несколькими экземплярами второго отношения, и каждый экземпляр второго отношения может быть связан с несколькими экземплярами первого отношения. Тип связи много-ко-многим является временным типом связи, допустимым на ранних этапах проектирования схемы базы данных.

Каждая связь может иметь одну из двух модальностей связи, таких как:

* «может»;
* «должен».

Модальность «может» означает, что экземпляр одного отношения может быть связан с одним или несколькими экземплярами другой отношения, а может быть и не связан ни с одним экземпляром.

Модальность «должен» означает, что экземпляр одного отношения обязан быть связан не менее чем с одним экземпляром другого отношения [6].

Далее при проектировании схемы базы данных следует определить атрибуты отношений.

Отношение «Профиль» имеет следующий набор атрибутов:

* идентификатор профиля;
* описание;
* город;
* веб сайт;
* номер телефона;
* пол;
* идентификатор записи;
* идентификатор пользователя

Отношение «Пользователь» имеет следующие атрибуты:

* идентификатор пользователя;
* пароль;
* логин;
* емеил;
* имя.
* фамилия;

Отношение «Запись» имеет следующие атрибуты:

* идентификатор записи;
* название;
* контент;
* время создания
* идентификатор автора;
* идентификатор лайка;
* идентификатор дислайка

Отношение «Друг» имеет следующие атрибуты:

* идентификатор друга;
* идентификатор пользователя;

Отношение «Лайк» имеет следующие атрибуты:

* идентификатор лайка;
* идентификатор записи;
* идентификатор пользователя;

Отношение «Дислайк» имеет следующие атрибуты:

* идентификатор лайка;
* идентификатор записи;
* идентификатор пользователя;

Отношение «Дислайк» имеет следующие атрибуты:

* идентификатор лайка;
* идентификатор записи;
* идентификатор пользователя;

Отношение «Сообщение» имеет следующие атрибуты:

* идентификатор сообщения;
* имя группы;
* контент сообщения;
* время отправления
* идентификатор пользователя

Процесс поуровневого спуска от наиболее общей и абстрактной концептуальной модели исходной бизнес-системы к логической, а затем и к физической модели соответствующей программной системы можно представить с использованием нотации UML.

Для достижения этих целей вначале строится модель в форме так называемой диаграммы вариантов использования (use case diagram), которая описывает функциональное назначение системы или, другими словами, то, что бизнес-система должна делать в процессе своего функционирования.

Основным назначением диаграммы вариантов использования, является описание функциональности и поведения приложения, позволяющее заказчику и разработчику совместно обсуждать проектируемую или существующую систему.

Назначение диаграммы вариантов использования состоит в следующем: проектируемая программная система представляется в форме так называемых вариантов использования, с которыми взаимодействуют внешние сущности или актеры. При этом актером или действующим лицом называется любой объект, субъект или система, взаимодействующая с моделируемой бизнес-системой извне. Это может быть человек, техническое устройство, программа или любая другая система, которая служит источником воздействия на моделируемую систему так, как определит разработчик. Вариант использования служит для описания сервисов, которые система предоставляет актеру. Другими словами, каждый вариант использования определяет набор действий, совершаемый системой при диалоге с актером. При этом ничего не говорится о том, каким образом будет реализовано взаимодействие актеров с системой и собственно выполнение вариантов использования.

Рассматривая диаграмму вариантов использования в качестве модели бизнес-системы, можно ассоциировать ее с «черным ящиком». Концептуальный характер этой диаграммы проявляется в том, что подробная детализация диаграммы или включение в нее элементов физического уровня представления на начальном этапе проектирования скорее имеет отрицательный характер, поскольку предопределяет способы реализации поведения системы. Эти аспекты должны быть сознательно скрыты от разработчика на диаграмме вариантов использования.

2.3 Выбор и обоснование инструментов разработки

Для разработки программного средства была выбрана система управления базами данных MySQL, язык программирования Python.

MySQL – одна из популярных СУБД, которая поддерживается этим языком.

Data Definition Language (DDL) (язык описания данных) - это язык, используемый в компьютерных  программах для описания структуры  баз данных.

Функции языков DDL определяются первым словом в предложении (часто называемом запросом), которое почти всегда является глаголом. В случае с SQL эти глаголы - "create" ("создать"), "alter" ("изменить"), "drop" ("удалить"). Это превращает природу языка в ряд обязательных утверждений (команд) к базе данных.

Data Manipulation Language (DML) (язык управления данными) - это язык, используемый в компьютерных программах или пользователями баз данных для получения, вставки, удаления или изменения данных в базах данных. Функции языков DML определяются первым словом в предложении (часто называемом запросом), которое почти всегда является глаголом. В случае с SQL эти глаголы - "insert" ("вставить"), "update" ("обновить"), и "delete" ("удалить"). Это превращает природу языка в ряд обязательных утверждений (команд) к базе данных.

Благодаря внутреннему механизму многопоточности быстродействие MySQL весьма высоко. Для разработчиков MySQL скорость всегда являлась ключевым параметром. Новые возможности добавлялись в пакет MySQL только после того, как их удавалось реализовать без ущерба для производительности. Иногда это означало, что некоторые возможности добавлялись не так быстро, как хотелось бы пользователям, но зато всегда гарантировало быструю работу MySQL.

Довольно высокий уровень безопасности обеспечивается благодаря базе данных MySQL, создающейся при установке пакета и содержащей пять таблиц. При помощи этих таблиц можно описать, какой пользователь из какого домена с какой таблицей может работать и какие команды он может применять. Пароли, хранящиеся в базе данных, можно зашифровать при помощи встроенной в MySQL функции password().

Благодаря открытости кода программист может сам добавлять в пакет нужные функции, расширяя его функциональность так, как ему требуется.

Для начала работы с MySQL не требуется сложной процедуры конфигурации. MySQL Server начнёт работать соответствующим образом сразу. По умолчанию выбираются значения, соответствующие минимальному использованию ресурсов диска и памяти. Для получения оптимальной производительности и для специальных условий (например, для проверки входа в систему), конечно же, потребуется дополнительная настройка. Чтобы помочь выполнить такую настройку, предлагаются соответствующие примеры файлов типовой конфигурации.

В настоящее время существуют версии программы для большинства распространенных компьютерных платформ. Это говорит о том, что вам не навязывают определенную операционную систему. Вы сами можете выбрать, с чем работать, например с Linux или Windows, но даже в случае замены ОС вы не потеряете свои данные, и вам даже не понадобятся дополнительные инструменты для их переноса.

Конечно же, как и любое программное средство, система управления базами данных MySQL не избавлена от некоторых недостатков. Например, можно назвать отсутствие вложенных запросов, что приводит к необходимости находить нужные значения отдельно и подставлять их в другой запрос непосредственно в CGI-сценарии, что, несомненно, сказывается на производительности.

Язык SQL имеет слудующие преимущества:

* независимость от конкретной СУБД. Несмотря на наличие диалектов и различий в синтаксисе, в большинстве своём тексты SQL-запросов, содержащие DDL и DML, могут быть достаточно легко перенесены из одной СУБД в другую. Существуют системы, разработчики которых изначально закладывались на применение по меньшей мере нескольких СУБД;
* наличие стандартов для выявления совместимости  и соответствия конкретной реализации SQL общепринятому стандарту только способствует «стабилизации» языка.
* декларативность. С помощью SQL программист описывает только то, какие данные нужно извлечь или модифицировать. То, каким образом это сделать, решает СУБД непосредственно при обработке SQL-запроса.

Недостатки:

* повторяющиеся строки;
* неопределённые значения (nulls);
* явное указание порядка колонок слева направо;
* колонки без имени и дублирующиеся имена колонок;
* отсутствие поддержки свойства «=»;
* использование указателей.

Для разработки пользовательского интерфейса был выбран язык Python и среда разработки Pycharm.

Язык программирования Python

Один из самых распространенных ЯП в современном мире, идеально подходящий для разработки Web приложений. Имеет большое количество встроенных библиотек для разработки сетевого ПО:

* Django (свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC);
* Flask (фреймворк для создания веб-приложений на языке программирования Python, использующий набор инструментов Werkzeug, а также шаблонизатор Jinja2.);
* Pygame (набор модулей (библиотек) языка программирования Python, предназначенный для написания компьютерных игр и мультимедиа-приложений.);
* Scrapy (бесплатная среда для сканирования веб-страниц с открытым исходным кодом.);
* Arrow (библиотека Python, которая предлагает разумный, понятный человеку подход к созданию, обработке, форматированию и преобразованию дат, времени и временных отметок.).

Удобный и легко читаемый код позволяет ориентироваться в коде, написанном другими пользователями.

IDE PyCharm удобная среда разработки для ЯП Python. Функции, предоставляемые ей:

* автоматическая подсветка ошибок кода или «неправильного» кода;
* автоматическое сохранение изменений;
* автоматическое подключение виртуальной среды для каждого проекта;
* выделение различных структур;
* автоматическое дополнение кода;
* одновременный запуск нескольких приложений из одного окна (в том числе и несколько запусков одного приложения).

ОС

Не имеет значения, какая операционная система будет применятся для использования данного ПО.

3 Программная реализация

3.1 Физическая структура

При проектировании разрабатываемого продукта была создана база данных состоящая из следующих таблиц:

* user;
* user\_profile;
* post;
* profile\_image;
* friend;
* like;
* dislike;
* message.

Таблица «user» хранит в себе базовые данные о пользователях. На рисунке 3 представлена данная таблица и её атрибуты.

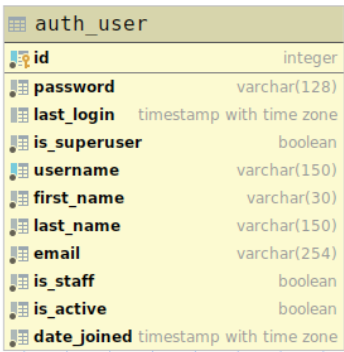


Рисунок 3 – Таблица «user»

Таблица «userprofile» хранит в дополнительную информацию о пользователе. На рисунке 4 представлена данная таблица и её атрибуты.

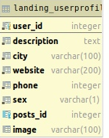


Рисунок 4 – Таблица «userprofile»

Таблица «post» хранит в информацию о содержимом статей. На рисунке 5 представлена данная таблица и её атрибуты.

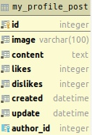


Рисунок 5 – Таблица «post»

Таблица «friend» хранит в информацию о друзьях пользователя. На рисунке 6 представлена данная таблица и её атрибуты.



Рисунок 6 – Таблица «friend»

Таблица «like» хранит в информацию о статусе оценки (положительная). На рисунке 7 представлена данная таблица и её атрибуты.



Рисунок 7 – Таблица «like»

Таблица «dislike» хранит в информацию о статусе оценки (отрицательная). На рисунке 7 представлена данная таблица и её атрибуты.

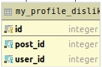


Рисунок 7 – Таблица «dislike»

Таблица «message» хранит в информацию о сообщениях конкретной группы пользователей. На рисунке 8 представлена данная таблица и её атрибуты.



Рисунок 8 – Таблица «message»

Таблица «profile\_image» хранит в информацию о фотографиях пользователя. На рисунке 9 представлена данная таблица и её атрибуты.

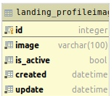


Рисунок 9 – Таблица «profile\_image»

В базе использовались следующие типы данных, приведенные ниже.

BOOLEAN: может хранить одно из двух значений: true или false.

INTEGER: хранит числа от -2147483648 до +2147483647. Занимает 4 байта.

VARCHAR: представляет строку из фиксированного количества символов.

CHAR: представляет строку из фиксированного количества символов.

BIGINT: хранит числа от -9223372036854775808 до +9223372036854775807.

В процессе логического проектирования данной базы данных была проведена нормализация всех отношений и таблица соответствует следующим критериям:

* отсутствие повторяющихся записей;
* отсутствие повторяющихся групп полей;
* атрибуты и строки не упорядочены;
* значения атрибутов атомарны;
* ни одно из не ключевых полей таблицы не идентифицируется с помощью другого не ключевого поля;
* любое не ключевое поле однозначно идентифицируется полным набором ключевых полей.

Следуя из вышеперечисленного можно сделать вывод о том, что база данных для данного программного продукта соответствует всем трём нормальным формам.

3.2 Описание разработанных модулей

При разработке программного продукта были выделены файлы описания стилей, форм:

- base.html – файл, описывающий макет разметки всех страниц;

- style.css – файл, позволяющий регулировать оформление блоков страницы (цвет, размер шрифта, отступы).

Так же были разработаны следующие классы:

- manage.py – файл для запуска приложения;

- LoginRequiredMixin – класс для авторизации пользователя. Содержит следующие методы:

1) LoginRequiredMixin.handle\_no\_permission() – метод для проверки прав пользователя к запрашиваемому объекту;

2) LoginRequiredMixin.get\_login\_url() – метод для получения url на страницу с авторизацией;

- UsersListView – класс для отображения пользователей зарегистрированных в системе;

- PostEditView – класс для редактирования записи от авторизованного пользователя. Содержит следующие методы:

1) PostEditView.get() – метод, который отправляет пользователю форму на редактирование записи;

2) PostEditView.port() – метод для обработки POST запросов, редактирует статью в соответствие с валидными данными, которые пришли на адрес записи;

- RegisterView – класс для создания пользователей от авторизованного пользователя. Содержит следующие методы:

1) RegisterView.get() – метод, который отправляет пользователю форму на регистрацию;

2) RegisterView.post() – метод для обработки POST запросов, создает запись нового пользователь в базе данных.

- Like/Dislike – модель таблицы «Like/Dislike» для оценок. Содержит следующие методы:

1) Like/Dislike.delete\_or\_switch() – метод для удаления и изменения оценки;

2) Like/Dislike.get\_related\_likes() – метод для получения количества всех положительных оценок к оцененному объекту;

3) Like/Dislike. get\_related\_dislikes() – метод для получения количества всех отрицательных оценок к оцененному объекту.

- MessageView() – модель таблицы «Message» хранения сообщений пользователей. Содержит следующие методы:

1) MessageView.post()– метод для обработки POST запросов;

2) MessageView.get()– метод для обработки GET запросов;

4 Тестирование

Процесс исследования, испытания программного обеспечения (ПО) с целью получения информации о качестве продукта является тестирование **программного обеспечения**.

Все виды тестирования программного обеспечения, в зависимости от преследуемых целей, можно условно разделить на функциональные и нефункциональные.

Функциональные виды тестирования — это тестирование программного продукта в целях проверки реализуемости функциональных требований, то есть его способности в определённых условиях решать задачи, нужные пользователям.

Тестирование функциональности может, проводится в двух аспектах: требования и бизнес-процессы.

Тестирование в перспективе «требования» использует спецификацию функциональных требований к системе как основу для дизайна тестовых случаев (Test Cases). В этом случае необходимо сделать список того, что будет тестироваться. Это позволит сфокусироваться и не упустить при тестировании наиболее важный функционал. Тестирование в перспективе «бизнес-процессы» использует знание бизнес-процессов, которые описывают сценарии ежедневного использования системы. В этой перспективе тестовые сценарии (test scripts), как правило, основываются на случаях использования системы (use cases). Преимуществом функционального тестирования является то, что оно имитирует фактическое использование системы.

Недостатки функционального тестирования:

– возможность упущения логических ошибок в программном обеспечении;

– вероятность избыточного тестирования.

Тестирование безопасности — это стратегия тестирования, используемая для проверки безопасности системы, а также для анализа рисков, связанных с обеспечением целостного подхода к защите приложения, атак взломщиков, вирусов, несанкционированного доступа к конфиденциальным данным.

Общая стратегия безопасности основывается на трех основных принципах:

– конфиденциальность;

– целостность;

– доступность.

Конфиденциальность — это сокрытие определенных ресурсов или информации. Под конфиденциальностью можно понимать ограничение доступа к ресурсу некоторой категории пользователей, или другими словами, при каких условиях пользователь авторизован получить доступ к данному ресурсу [3].

Доступность представляет собой требования о том, что ресурсы должны быть доступны авторизованному пользователю, внутреннему объекту или устройству. Как правило, чем более критичен ресурс, тем выше уровень доступности должен быть.

В настоящее время наиболее распространенными видами уязвимости в безопасности программного обеспечения являются вставки скриптов – Code injections (SQL, PHP, ASP) ­— это вид уязвимости, при котором становится возможно осуществить запуск исполняемого кода с целью получения доступа к системным ресурсам, несанкционированного доступа к данным либо выведения системы из строя.

Нефункциональное тестирование описывает тесты, необходимые для определения характеристик программного обеспечения, которые могут быть измерены различными величинами. В целом, это тестирование того, как система работает.

Задачей тестирования производительности является определение масштабируемости приложения под нагрузкой, при этом происходит измерение времени выполнения выбранных операций при определенных интенсивностях выполнения этих операций, определение количества пользователей, одновременно работающих с приложением, определение границ приемлемой производительности при увеличении нагрузки (при увеличении интенсивности выполнения этих операций), исследование производительности на высоких, предельных, стрессовых нагрузках.

Стрессовое тестирование позволяет проверить насколько приложение, и система в целом работоспособны в условиях стресса и также оценить способность системы к регенерации, то есть к возвращению к нормальному состоянию после прекращения воздействия стресса. Стрессом в данном контексте может быть повышение интенсивности выполнения операций до очень высоких значений или аварийное изменение конфигурации сервера. Также одной из задач при стрессовом тестировании может быть оценка деградации производительности, таким образом цели стрессового тестирования могут пересекаться с целями тестирования производительности.

Задачей объемного тестирования является получение оценки производительности при увеличении объемов данных в базе данных приложения, при этом происходит измерение времени выполнения выбранных операций при определенных интенсивностях выполнения этих операций, может производиться определение количества пользователей, одновременно работающих с приложением.

Задачей тестирования стабильности (надежности) является проверка работоспособности приложения при длительном (многочасовом) тестировании со средним уровнем нагрузки. Время выполнения операций может играть в данном виде тестирования второстепенную роль. При этом на первое место выходит отсутствие утечек памяти, перезапусков серверов под нагрузкой и другие аспекты влияющие именно на стабильность работы.

Тестирование удобства пользования — это метод тестирования, направленный на установление степени удобства использования, обучаемости, понятности и привлекательности для пользователей разрабатываемого продукта в контексте заданных условий.

Тестирование удобства пользования дает оценку уровня удобства использования приложения по следующим пунктам: производительность; эффективность (efficiency) — сколько времени и шагов понадобится пользователю для завершения основных задач приложения, например, размещение новости, регистрации, покупка и так далее (меньше — лучше); правильность (accuracy) — сколько ошибок сделал пользователь во время работы с приложением (меньше — лучше); активизация в памяти (recall) — как много пользователь помнит о работе приложения после приостановки работы с ним на длительный период времени (повторное выполнение операций после перерыва должно проходить быстрее чем у нового пользователя).

Эмоциональная реакция (emotional response) — как пользователь себя чувствует после завершения задачи — растерян или испытал стресс.

Проверка удобства использования может проводиться как по отношению к готовому продукту, посредством тестирования черного ящика (black box testing), так и к интерфейсам приложения (API), используемым при разработке — тестирование белого ящика (white box testing). В этом случае проверяется удобство использования внутренних объектов, классов, методов и переменных, а также рассматривается удобство изменения, расширения системы и интеграции ее с другими модулями или системами. Использование удобных интерфейсов (API) может улучшить качество, увеличить скорость написания и поддержки разрабатываемого кода, и как следствие улучшить качество продукта в целом.

Отсюда становится очевидно, что тестирование удобства пользования может производиться на разных уровнях разработки программного обеспечения: модульном, интеграционном, системном и приемочном. При этом оно целиком и полностью будет зависит от того, кто будет использовать приложение на выделенном конкретном уровне — разработчик, бизнес пользователь системы и так далее.

Для дизайна удобных приложений полезно следовать принципам fail-safe. Простой пример, если поле требует цифровое значение, логично ограничить пользователю диапазон ввода только цифрами — будет меньше случайных ошибок.

Методика подобного тестирования заключается в симулировании различных условий сбоя и последующем изучении и оценке реакции защитных систем. В процессе подобных проверок выясняется, была ли достигнута требуемая степень восстановления системы после возникновения сбоя.

Для наглядности рассмотрим некоторые варианты подобного тестирования и общие методы их проведения. Объектом тестирования в большинстве случаев являются весьма вероятные эксплуатационные проблемы, такие как:

– отказ электричества на компьютере-сервере;

– отказ электричества на компьютере-клиенте;

– незавершенные циклы обработки данных (прерывание работы фильтров данных, прерывание синхронизации);

– объявление или внесение в массивы данных невозможных или ошибочных элементов;

– отказ носителей данных.

Данные ситуации могут быть воспроизведены, как только достигнута некоторая точка в разработке, когда все системы восстановления или дублирования готовы выполнять свои функции. Технически реализовать тесты можно следующими путями:

* симулировать внезапный отказ электричества на компьютере (обесточить компьютер);
* симулировать потерю связи с сетью (выключить сетевой кабель, обесточить сетевое устройство);
* симулировать отказ носителей (обесточить внешний носитель данных);
* симулировать ситуацию наличия в системе неверных данных (специальный тестовый набор или база данных).

При достижении соответствующих условий сбоя и по результатам работы систем восстановления, можно оценить продукт с точки зрения тестирования на отказ. Во всех вышеперечисленных случаях по завершении процедур восстановления должно быть достигнуто определенное требуемое состояние данных.

Конфигурационное тестирование (Configuration Testing) — специальный вид тестирования, направленный на проверку работы программного обеспечения при различных конфигурациях системы (заявленных платформах, поддерживаемых драйверах, при различных конфигурациях компьютеров и так далее).

Для клиент-серверных приложений конфигурационное тестирование можно условно разделить на два уровня (для некоторых типов приложений может быть актуален только один): серверный и клиентский. На первом (серверном) уровне, тестируется взаимодействие выпускаемого программного обеспечения с окружением, в которое оно будет установлено.

Основной упор здесь делается на тестирование с целью определения оптимальной конфигурации оборудования, удовлетворяющего требуемым характеристикам качества (эффективность, портативность, удобство сопровождения, надежность).

На следующем (клиентском) уровне, программное обеспечение тестируется с позиции его конечного пользователя и конфигурации его рабочей станции. На этом этапе будут протестированы следующие характеристики: удобство использования, функциональность. Для этого необходимо будет провести ряд тестов с различными конфигурациями рабочих станций.

Уже на начальном этапе становится очевидно, что чем больше требований к работе приложения при различных конфигурациях рабочих станций, тем больше тестов необходимо будет провести. В связи с этим, рекомендуется, по возможности, автоматизировать этот процесс, так как именно при конфигурационном тестировании автоматизация реально помогает сэкономить время и ресурсы. Конечно же, автоматизированное тестирование не является панацеей, но в данном случае оно окажется очень эффективным помощником.

Тестирование, направленное на определение соответствия выпущенной версии, критериям качества по своим целям является аналогом дымового тестирования, направленного на приемку новой версии в дальнейшее тестирование или эксплуатацию. Вглубь оно может проникать дальше, в зависимости от требований к качеству выпущенной версии.

Существуют специальные методы для тестирования аспектов программ, не являющихся функциональными, то есть не относящихся к работоспособности самих программ, например, тестирование интерфейса пользователя, его удобности, практичности и лёгкости для освоения обычным пользователем.

При выборе способа тестирования разработанного программного средства для комплексного тестирования программы были применены методы тестирования по стратегии «Черного ящика», потому что они имеют цель выяснить причины, в которых поведение программы не соответствует указанному составу выполняемых функций требованиям.

При проведении тестирования была проверена работа всех ссылок, размещенных на сайте. Все переходы осуществляются корректно в соответствии с междустраничными связями. Ссылки, которые имеют не существующий адрес, отсутствуют.

Второй этап тестирования заключался в тестах на удобство пользования — эти тесты помогают выявить трудности, возникающие у пользователя при работе с программой, а также определить причины этих затруднений [8].

В ходе тестирования было выявлено, что программный продукт является вполне удобным для пользователя, так как позволяет беспрепятственно переходить с одной страницы на другую без возвращения на предыдущие страницы, а также обладает интуитивно понятным интерфейсом, даже для пользователя, который зашел на него первый раз. Сборка сайта при тестировании осуществлялась пошаговым методом сверху - вниз. Такой метод сборки предполагает последовательное подключение к набору уже оттестированных страниц очередной страницы.

Во время второго этапа тестирования был составлен ряд тестовых заданий, которые система должна выполнить корректно. Список заданий представлен в таблице 4.1.

Таблица 4.1 — Результаты выполнения тестовых заданий

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название операции | Событие | | Ожидаемый результат | | Фактический результат |
| 1 | 2 | | 3 | | 4 |
| Переход на главную страницу | Загрузка главной страницы | | Переход на главную страницу | | Переход на главную страницу |
| Просмотр отображаемых записей | Загрузка отображаемых записей | | Просмотр отображаемых записей | | Просмотр отображаемых записей |
| Добавление записи | Ввод текста записи и нажатие на кнопку добавления записи | | Добавление записи в базу данных и отображение на странице | | Добавление записи в базу данных и отображение на странице |
| Редактирование записи | Изменение текста записи и нажатие на кнопку редактирования записи | | Добавление измененной записи в базу данных и отображение на странице | | Добавление измененной записи в базу данных и отображение на странице |
| Добавление друга | Нажатие на кнопку добавления друга | | Добавление пользователя в друзья | | Добавление пользователя в друзья |
| Удаление друга | Нажатие на кнопку удаления друга | | Удаление пользователя из друзей | | пользователя |
| Продолжение таблицы 4.1 | | | | | |
| 1 | 2 | 3 | | 4 | |
| Положительная оценка записи | Нажатие на положительную оценку записи | Счетчик положительных оценок увеличился на 1 | | Счетчик положительных оценок увеличился на 1 | |
| Отрицательная оценка записи | Нажатие на отрицательную оценку записи | Счетчик отрицательных оценок увеличился на 1 | | Счетчик отрицательных оценок увеличился на 1 | |
| Авторизация пользователя | Заполнение форм авторизации и нажатие на кнопку войти | Пользователь авторизирован | | Пользователь авторизирован | |
| Регистрация пользователя | Заполнение форм регистрации и нажатие на кнопку регистрация | Пользователь зарегистрирован | | Пользователь зарегистрирован | |
| Написать сообщение | Заполнение формы сообщения и нажатие на кнопку отправить сообщение | Сообщение отправлено и появилось в базе данных | | Сообщение отправлено и появилось в базе данных | |
| Редактирование профиля | Нажатие на редактировать и заполнение выбранных полей | Профиль отредактирован | | Профиль отредактирован | |

Все тесты производились как в браузере, так и на мобильном устройстве. После проведения тестовых заданий над программным средством была выявлена некорректная работа системы. Все недостатки были исправлены, система имеет стабильное состояние.

5 Применение

5.1 Назначение и условия применения

Существенное преимущество построения веб-приложений для поддержки стандартных функций браузера заключается в том, что функции выполняется независимо от операционной системы данного клиента. Вместо того, чтобы писать различные версии для Microsoft Windows, Mac OS X, GNU/Linux и других операционных систем, приложение создаётся один раз для произвольно выбранной платформы и на ней разворачивается.

Однако различная реализация HTML, CSS и других спецификаций в браузерах может вызвать проблемы при разработке веб-приложений и последующей поддержке. Кроме того, возможность пользователя настраивать многие параметры браузера (например, размер шрифта, цвета, отключение поддержки сценариев) может препятствовать корректной работе приложения.

Для корректной работы пользователя с данными социальной сети браузер должен соответствовать критериям, описанным в таблице 1.

Таблица 1 – Перечень требуемых версий браузеров

|  |  |
| --- | --- |
| Браузер | Версия |
| Firefox | Не старше 20.0 |
| Chrome | Не старше 29.0 |
| Internet Explorer | Не старше11.0 |
| Opera | Не старше 12.10 |
| Safari | Не старше 6.1 |

Также для работы с данным веб-приложением требуется доступ пользовательского компьютера к сети Интернет.

Назначением данного интернет-магазина является продажа радиоуправляемой техники в сети Интернет, информация о которой хранится в базе данных, а также ознакомление клиентов с предоставляемыми товарами.

5.2 Руководство пользователя

Для запуска приложения необходимо перейти по ссылке «http://localhost:8000/

accounts/register».

После запуска будет открыта страница, на которой будет отображена форма регистрации. Пример окна предоставляемого эту информацию представлен на рисунке 5.1.

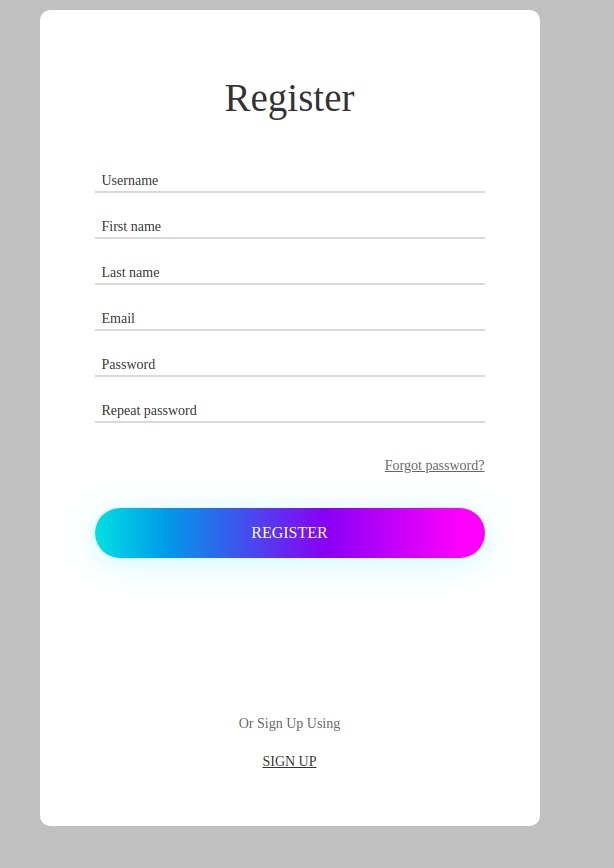


Рисунок 5.1 – Форма регистрации

После регистрации произойдет переход на страницу авторизации. Пример окна предоставляемого эту информацию представлен на рисунке 5.2.

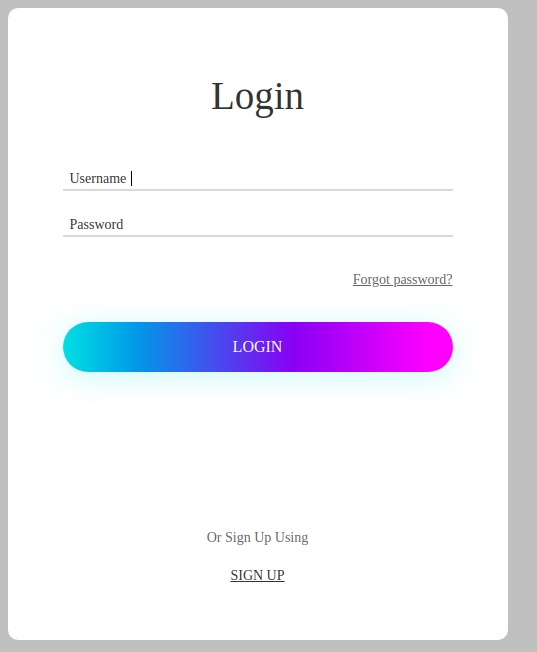


Рисунок 5.2 – Форма авторизации

После входа в аккаунт отобразится профиль пользователя. Пример окна предоставляемого эту информацию представлен на рисунке 5.3.



Рисунок 5.3 – Страница пользователя

Если нажать на «edit profile», то откроется страница редактирования профиля. Пример окна представлен на рисунке 5.4.

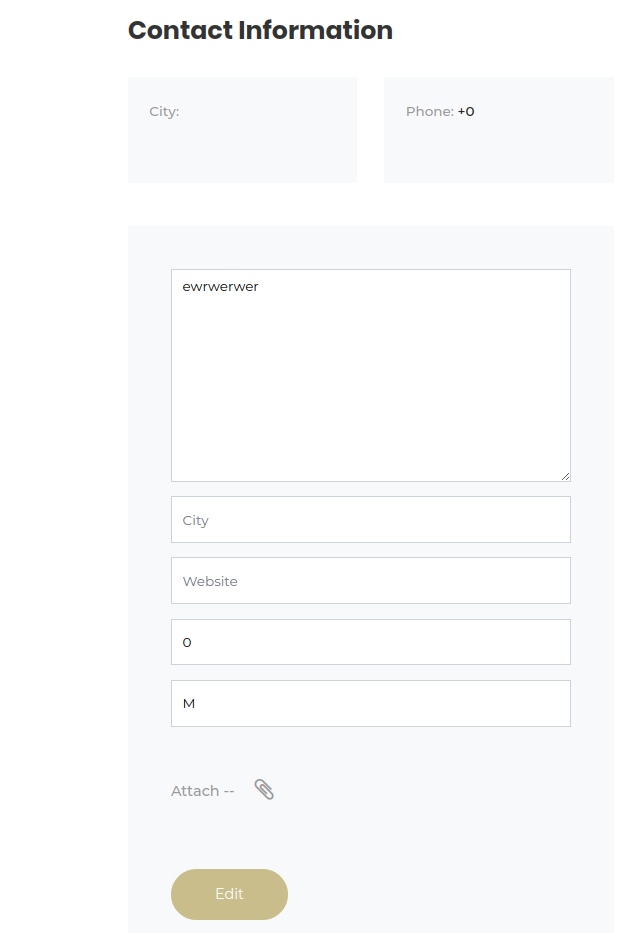


Рисунок 5.4 – Форма редактирования

Чтобы создать запись нужно перейти на страницу профиля и в форму снизу страницы ввести данные. Пример представлен на рисунке 5.5.

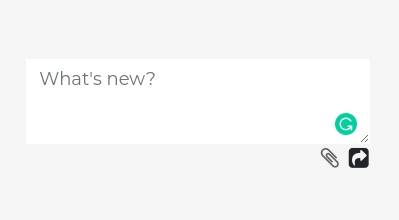


Рисунок 5.5 – форма создания записи

Чтобы посмотреть всех пользователей, зарегистрированных в системе нужно нажать «peoples» на главной странице аккаунта. Пример представлен на рисунке 5.6.

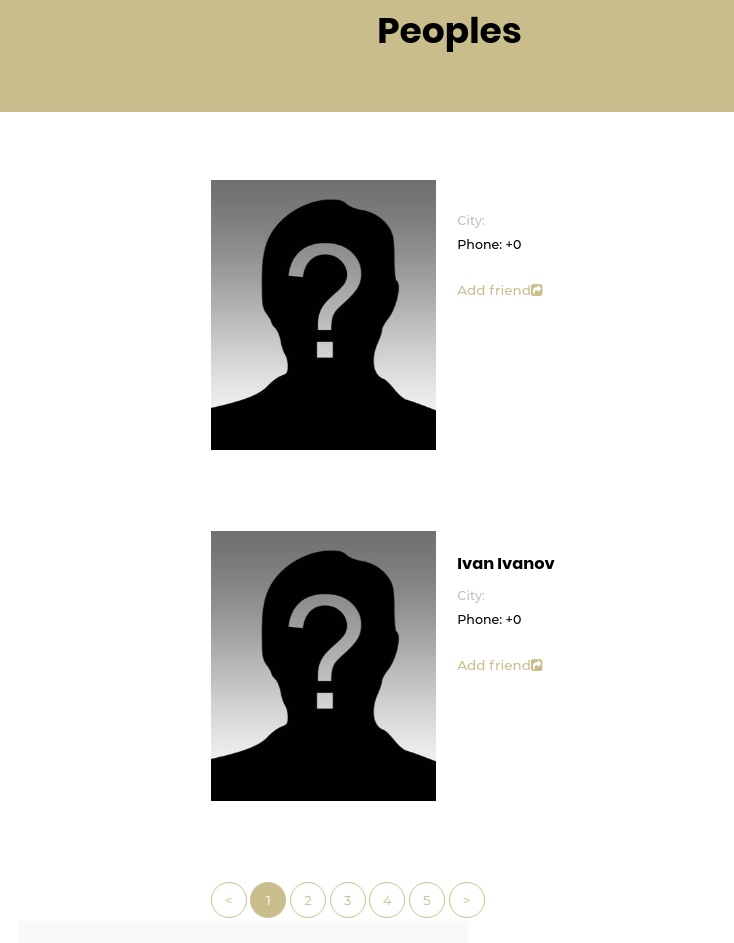


Рисунок 5.6 – страница пользователей

Чтобы отправить сообщение, нужно нажать кнопку отправить сообщение. После появится форма чата, которая представлена на рисунке 5.7.

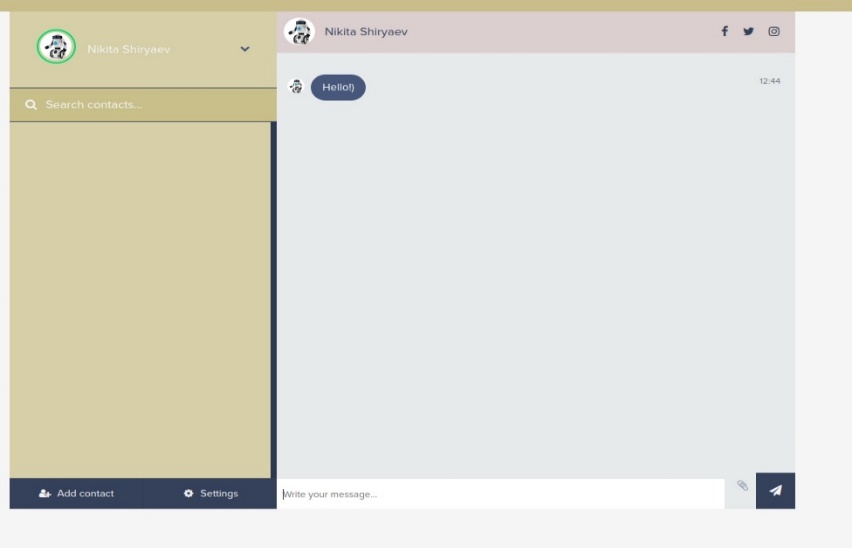


Рисунок 5.7 – форма чата

В данном разделе приведен весь процесс работы с программным средством.

6 Организация пожаробезопасных условий труда при эксплуатации ЭВМ в процессе использования социальной сети с возможностью передачи файлов "TalkIn"

6.1 Характеристика источников ультразвука в проектируемом объекте

Излучатели ультразвука - устройства, применяемые для возбуждения ультразвуковых колебаний и волн в газообразных, жидких и твердых средах. Излучатели ультразвука преобразуют в энергию [звукового поля](http://engineering-solutions.ru/ultrasound/theory/#field) энергию какого-либо другого вида.

Наибольшее распространение в качестве излучателей ультразвука получили электроакустические преобразователи. В подавляющем большинстве излучателей ультразвука этого типа, а именно в [пьезоэлектрических преобразователях](http://engineering-solutions.ru/ultrasound/transducers/), магнитострикционных преобразователях, электродинамических излучателях, электромагнитных и электростатических излучателях, электрическая энергия преобразуется в энергию колебаний какого-либо твердого тела (излучающей пластинки, стержня, диафрагмы и т.п.), которое и излучает в окружающую среду акустические волны. Все перечисленные преобразователи, как правило, линейны, и, следовательно, колебания излучающей системы воспроизводят по форме возбуждающий электрический сигнал; лишь при очень больших амплитудах колебаний вблизи верхней границы динамического диапазона излучателя ультразвука могут возникнуть нелинейные искажения.

В преобразователях, предназначенных для излучения монохроматической волны, используется явление резонанса: они работают на одном из собственных колебаний механической колебательной системы, на частоту которого настраивается генератор электрических колебаний, возбуждающий преобразователь. Электроакустические преобразователи, не обладающие твердотельной излучающей системой, применяются в качестве излучателей ультразвука сравнительно редко; к ним относятся, например, излучатели ультразвука, основанные на электрическом разряде в жидкости или на электрострикции жидкости.

К основным характеристикам излучателей ультразвука относятся их частотный спектр, излучаемая мощность звука, направленность излучения. В случае моночастотного излучения основными характеристиками являются рабочая частота излучателя ультразвука и его частотная полоса, границы которой определяются падением излучаемой мощности в два раза по сравнению с её значением на частоте максимального излучения

Излучатели ультразвука (электроакустические преобразователи) характеризуются чувствительностью, электроакустическим коэффициентом полезного действия и собственным электрическим импедансом.

Чувствительность излучателя ультразвука - отношение звукового давления в максимуме характеристики направленности на определённом расстоянии от излучателя (чаще всего на расстоянии 1 м) к электрическому напряжению на нём или к протекающему в нём току. Эта характеристика применяется к излучателям ультразвука, используемым в системах звуковой сигнализации, в гидролокации и в других подобных устройствах. Для излучателей технологического назначения, применяемых, например, при ультразвуковых очистке, коагуляции, воздействии на химические процессы, основной характеристикой является мощность.

Эффективность электроакустических преобразователей, излучающих акустическую энергию в озвучиваемую среду, характеризуют величиной их электроакустического коэффициента полезного действия. Для характеристики преобразования энергии в излучателях ультразвука используют эффективный коэффициент электромеханической связи.[9]

6.2 Средства индивидуальной и коллективной защиты от ультразвукового излучения

Коллективная защита от воздействия ультразвука: уменьшение вредного излучения ультразвуковой энергии в источнике ее возникновения; локализация действия ультразвука конструктивными и планировочными решениями; проведение организационно-профилактических мероприятий, а именно: дистанционное управление, автоблокировка, экранирование (использование кожухов), облицовка помещений звукопоглощающими материалами, проведение предварительных и периодических медицинских осмотров.

В качестве средств индивидуальной защиты используются противошумные наушники, вкладыши, защитные рукавицы и перчатки.

Предусматривают защиту от шума строительно-акустическими методами:

- звукоизоляция ограждающий конструкций; уплотнение по периметру притворов окон, ворот, дверей; звукоизоляция мест пересечения ограждающий конструкций инженерными коммуникациями; устройство звукоизолированных кабин наблюдения и дистанционного управления технологическим оборудование; укрытия и кожухи источников шума;

- установка в помещениях звукопоглощающих конструкций и экранов;

- применение глушителей аэродинамического шума, звукопоглощающей облицовки и газовоздушных трактах вентиляционных систем и механическим побуждением и систем кондиционирования воздуха;

В качестве индивидуальных средств защиты от шума используют специальные наушники, которые должны плотно закрывать уши, беруши должны быть подобраны по размеру слухового прохода, противошумные каски, защитное действие которых основано на изоляции и поглощении звуков.[11]

**Требования по ограничению неблагоприятного влияния ультразвука на работающих включают следующее:**

- запрещается непосредственный контакт человека с рабочей поверхностью источника ультразвука и с контактной средой. Для защиты рук от неблагоприятного воздействия контактного ультразвука в твердых, жидких, газообразных средах необходимо применять нарукавники, рукавицы или перчатки (наружные резиновые и внутренние хлопчатобумажные);

- при систематической работе с источниками контактного ультразвука в течение более 50% рабочего времени необходимо устраивать два регламентированных перерыва - десятиминутный перерыв за 1-1,5 часа до и пятнадцатиминутный перерыв через 1,5-2 часа после обеденного перерыва для проведения физиопрофилактических процедур (тепловых гидропроцедур, массажа, ультрафиолетового облучения), а также лечебной гимнастики, витаминизации и т.п.;

Организационно-профилактические мероприятия заключаются в проведении инструктажа и установлении рациональных режимов труда и отдыха. К работе с ультразвуковыми источниками допускаются лица не моложе 18 лет, прошедшие соответствующий курс обучения. Лица, подвергающиеся в процессе трудовой деятельности воздействию контактного ультразвука, подлежат предварительным, при приеме на работу, и периодическим медицинским осмотрам.[12]

Снижение неблагоприятного воздействия инфразвука достигается комплексом инженерно-технических и медицинских мероприятий, из которых основными являются: ослабление инфразвука в его источнике, устранение причин воздействия; изоляцию инфразвука; поглощение инфразвука, постановку глушителей; индивидуальные средства защиты; медицинскую профилактику. Борьба с неблагоприятным воздействием инфразвука должна вестись в тех же направлениях, что и борьба с шумом. Наиболее целесообразно уменьшать интенсивность инфразвуковых колебаний на стадии проектирования машин или агрегатов. Первостепенное значение в борьбе с инфразвуком имеют методы, снижающие его возникновение и ослабление в источнике.[11]

6.3 Выбор и обоснование защиты работающих от ультразвукового излучения при работе с проектируемым объектом

Для защиты от низких инфразвуковых частот звукоизоляция крайне неэффективна - требуются очень толстые и массивные звукоизолирующие перегородки. Также неэффективны звукопоглощение и акустическая обработка помещений. Поэтому основным методом борьбы с инфразвуком является борьба в источнике его возникновения.

Другими мероприятиями по борьбе с инфразвуком являются:

- повышение быстроходности машин, что обеспечивает перевод максимума излучения в область слышимых частот, где становятся эффективными звукоизоляция и звукопоглощение;

- устранение низкочастотных вибраций;

- применение глушителей реактивного типа.

Ультразвук из-за очень высоких частот быстро поглощается в воздухе и материалах конструкций, поэтому он распространяется на небольшие расстояния. Для защиты от ультразвука очень эффективной является звукоизоляция и звукопоглощение. Обычно источники ультразвука заключают в кожухи из тонкой стали, алюминия (толщиной 1 мм), обклеенные внутри резиной. Применяют также эластичные кожухи из нескольких слоев резины общей толщиной 3,5 мм. Эффективность таких кожухов может достигать 60...80 дБ. Применяют также экраны, расположенные между источником и работающими.

Экранирование излучений. Экранируют либо источники излучения, либо зоны, где может находиться человек. Экраны могут быть замкнутыми (полностью изолирующими излучающее устройство или защищаемый объект) или незамкнутыми, различной формы и размеров, выполненными из сплошных, перфорированных, сотовых или сетчатых материалов. Экраны частично отражают и частично поглощают электромагнитную энергию. По степени отражения и поглощения их условно разделяют на отражающие и поглощающие экраны.

Отражающие экраны выполняют из хорошо проводящих материалов, например стали, меди, алюминия толщиной не менее 0,5 мм из конструктивных и прочностных соображений. Кроме сплошных, перфорированных, сетчатых и сотовых экранов могут применяться: фольга, наклеиваемая на несущее основание; токопроводящие краски (для повышения проводимости красок в них добавляют порошки коллоидного серебра, графита, сажи, окислов металлов, меди, алюминия), которыми окрашивают экранирующие поверхности; экраны с металлизированной со стороны падающей электромагнитной волны поверхностью.

Поглощающие экраны выполняют из радиопоглощающих материалов. Есественных материалов с хорошей радиопоглощающей способностью нет, поэтому их выполняют с помощью различных конструктивных приемов и введением различных поглощающих добавок в основу. В качестве основы используют каучук, поролон, пенополистирол, пенопласт, керамикометаллические композиции и т.д. В качестве добавок применяют сажу, активированный уголь, порошок карбонильного железа и пр. Все экраны обязательно должны заземляться для обеспечения стекания образующихся на них зарядов в землю.

Для увеличения поглощающей способности экрана их делают многослойными и большой толщины, иногда со стороны падающей волны выполняют конусообразные выступы.[11]

Наиболее часто в технике защиты от электромагнитных полей применяют металлические сетки. Они легки, прозрачны, поэтому обеспечивают возможность наблюдения за технологическим процессом и излучателем, пропускают воздух, обеспечивая охлаждение оборудования за счет естественной или искусственной вентиляции.

7 Экономический раздел

# 7.1 Технико-экономическое обоснование разработки cоциальной сети с возможностью передачи файлов «TalkIn»

# 7.1.1 Исходные данные

Исходные данные для расчета приведены в таблице 7.1.

Таблица 7.1 – Исходные данные

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование показателя | Единица измерения | Условные обозначения | Норматив |
| 1 | 2 | 3 | 4 |
| Коэффициент изменения скорости обработки информации | ед. |  | 0,5 |
| Численность разработчиков | чел. |  | 1 |
| Базовая ставка 1-го разряда в организации | руб. |  | 185 |
| Тарифный коэффициент | ед. |  | 1,29 |
| Фонд рабочего времени | Ч |  | 2032 |
| Коэффициент естественных потерь рабочего времени | ед. |  | 1,2 |
| Коэффициент премирования | ед. |  | 1,2 |
| Норматив дополнительной заработной платы | % |  | 15 |
| Ставка отчислений в фонд социальной защиты населения | % |  | 34 |
| Ставка отчислений по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний | % |  | 0,6 |
| Цена одного машино-часа | руб. |  | 2,50 |
| Норматив прочих затрат | % |  | 20 |
| Норматив накладных расходов | % |  | 100 |
| Норматив расходов на сопровождение и адаптацию | % |  | 10 |
| Уровень рентабельности | % |  | 20 |
| Ставка НДС | % |  | 20 |
| Норматив расходов на освоение ПС | % |  | 1 |

Продолжение таблицы 6.1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Норматив расходов на пополнение оборотных средств в связи с использованием нового ПС | % |  | 2 |
| Ставка налога на прибыль | % |  | 18 |
| Норматив приведения разновременных затрат | ед. |  | 0,09 |

# 

# 7.1.2 Расчет цены и прибыли на ПС

В современных рыночных экономических условиях ПС выступает преимущественно в виде продукции организаций, представляющей собой функционально завершенные и имеющие товарный вид ПС ВТ, реализуемые покупателям по рыночным отпускным ценам. Все завершенные разработки ПС ВТ являются научно-технической продукцией.

Широкое применение ВТ требует постоянного обновления и совершенствования ПС. Выбор эффективных проектов ПС связан с их экономической оценкой и расчетом экономического эффекта, который может определяться как у разработчика, так и у пользователя.

У разработчика экономический эффект выступает в виде прибыли от реализации ПС, остающейся в распоряжении организации, а у пользователя – в виде экономии трудовых, материальных и финансовых ресурсов, получаемой за счет:

* снижения трудоемкости расчетов и алгоритмизации программирования и отладки программ за счет использования ПС в процессе разработки автоматизированных систем обработки данных;
* сокращения расходов на оплату машинного времени и других ресурсов на отладку программ;
* снижения расходов на материалы (магнитные ленты, магнитные диски и прочие материалы);
* ускорения ввода в эксплуатацию новых систем;
* улучшения показателей основной деятельности в результате использования ПС.

Стоимостная оценка ПС у разработчиков предполагает определение затрат, что включает следующие статьи:

* заработная плата исполнителей – основная и дополнительная;
* отчисления в фонд социальной защиты населения;
* отчисления по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний;
* расходы на материалы и комплектующие;
* расходы на спецоборудование;
* расходы на оплату машинного времени;
* прочие прямые затраты;
* накладные расходы.

На основании затрат рассчитывается себестоимость и отпускная цена ПС.

# 7.1.2.1 Общая характеристика разрабатываемого ПС ВТ

# 

Программный продукт: «Социальная сеть с возможностью передачи файлов «TalkIn»». Является сервисом для общения пользователей и обмена файлами. Состоит из базы данных, в которой хранятся данные пользователях, комментариях и постах.

Выполняемые функции:

* обеспечение интерфейса между компонентами;
* организация ввода информации;
* организация поиска и поиск в базе данных;
* формирование баз данных;
* графический вывод результатов;
* организация ввода/вывода информации в интерактивном режиме;
* мониторинг ПС ВТ.

Тип программного средства: программное средство общего назначения.

Среда разработки приложения: Python, Windows.

Разработка программного средство производится в среде IntelliJ IDEA на языке программирования Python. База данных PosgreSQL.

Так как ПС ВТ обладает следующей характеристикой: режим работы в реальном времени – группа сложности ПС ВТ 1-ая.

Поскольку программное средство является развитием определенного параметрического ряда программного средства вычислительной техники, то из таблицы «Поправочные коэффициенты, учитывающие новизну ПС ВТ» следует, что оно относится к группе Б.

Рынок на данный момент представлен рядом аналогов и конкурентов: Vk.com, Facebook.com, twitter.com. Мой ресурс работает без прерываний, так как информация в базе данных динамически сортируется и удаляется повторная ненужная информация.

# 7.1.2.2 Определение объема ПС ВТ

Объем ПС определяется путем подбора аналогов на основании классификации типов ПС, каталога функций ПС и аналогов ПС в разрезе функций, которые постоянно обновляются и утверждаются в установленном порядке.

На основании информации о функциях разрабатываемого ПС по каталогу функций определяется объем функций. Общий объем ПС рассчитывается по формуле (6.1).

|  |  |
| --- | --- |
|  | (7.1) |

где – общий объем ПС, условные машино-команды;

– общее число функций;

– объем-й функции ПС, условные машино-команды.

(машино-команд)

Данные размещены в таблице 6.2.

Таблица 7.2 – Содержание и объем функций на разрабатываемое ПС ВТ

|  |  |
| --- | --- |
| Содержание функции | Объем, условных машино-команд |
| Организация ввода информации | 13 |
| Организация поиска и поиск в базе данных | 553 |
| Графический вывод результатов | 50 |
| Обеспечение интерфейса между компонентами | 10 |
| Преобразование операторов входного языка и команды другого языка | 90 |
| Формирование баз данных | 230 |
| Организация ввода/вывода информации в интерактивном режиме | 31 |
| Мониторинг ПС ВТ | 65 |
| Итого | 1042 |

В связи с быстрым изменением ВТ необходимо определить скорректированный объем функций по формуле (7.2).

|  |  |
| --- | --- |
|  | (6.2) |

где – скорректированный объем ПС, условные машино-команды;

– общий объем ПС, условные машино-команды;

– коэффициент изменения скорости обработки информации.

(машино-команд)

# 7.1.2.3 Расчет трудоемкости выполняемой работы

Определение трудоемкости необходимо для дальнейшего расчета суммы основной заработной платы.

Сумма основной заработной платы рассчитывается на основе численности специалистов, соответствующих тарифных ставок и фонда рабочего времени. Причем численность специалистов, календарные сроки разработки программы и фонд рабочего времени определяются по укрупненным нормам времени на разработку, сопровождение и адаптацию ПС), или экспертным путем. Расчет трудоемкости ПС с использованием укрупненных норм времени осуществляется в основном в крупных научно-технических организациях для решения сложных задач программного обеспечения ВТ. В мелких и средних научно-технических организациях трудоемкость, численность исполнителей и сроки разработки ПС определяются экспертным путем с использованием данных по базовым моделям.

При определении трудоемкости ПС учитываются объем ПС (в тысячах строк условного кода), объем документации (тысячах строк), новизна и сложность ПС, язык программирования, степень использования типовых (стандартных программ).

Нормативная трудоемкость устанавливается с учетом сложности ПС. Выделяется три группы сложности, в которых учтены следующие составляющие ПС: языковой интерфейса, ввод-вывод, организация данных, режим работы, операционная и техническая среда. Кроме того, устанавливаются дополнительные коэффициенты сложности ПС.

С учетом дополнительного коэффициента сложности рассчитывается общая трудоемкость ПС по формуле (6.3).

|  |  |
| --- | --- |
|  | (7.3) |

где – общая трудоемкость ПС, человеко-дней;

– нормативная трудоемкость ПС, человеко-дней;

– дополнительный коэффициент сложности ПС, ед.

(чел.-дн)

При решении сложных задач с длительным периодом разработки ПС трудоемкость определяется по стадиям разработки (техническое задание – ТЗ, эскизный проект – ЭП, технический проект – ТП, рабочий проект – РП и внедрение – ВН) с учетом новизны, степени использования типовых программ и удельного веса трудоемкости стадий разработки ПС и общей трудоемкости разработки ПС. Методика определения трудоемкости в данном случае несколько иная.

# 7.1.2.4 Расчет основной заработной платы

В соответствии с «Рекомендациями по применению «тарифной сетки работников Республики Беларусь» и тарифными разрядами и коэффициентами должностей» каждому исполнителю устанавливается разряд и тарифный коэффициент.

Месячная базовая ставка каждого исполнителя определяется путем умножения действующей месячной базовая ставки 1-го разряда на тарифный коэффициент, соответствующий установленному тарифному разряду, и рассчитывается по формуле (6.4).

|  |  |
| --- | --- |
|  | (7.4) |

где – тарифная ставка за месяц, руб.;

– базовая ставка 1-го разряда за месяц, руб.;

– тарифный коэффициент, ед.

(руб.)

Основная заработная плата исполнителей на конкретное ПС рассчитывается по формуле (6.5).

|  |  |
| --- | --- |
|  | (7.5) |

где – основная заработная плата, руб.;

– тарифная ставка за день, руб.;

– общая трудоемкость ПС, человеко-дней;

– коэффициент естественных потерь рабочего времени, ед.;

– коэффициент премирования.

# 7.1.2.5 Расчет дополнительной заработной платы

# 

Дополнительная заработная плата на конкретное ПС включает выплаты, предусмотренные законодательством о труде (оплата отпусков, льготных часов, времени выполнения государственных обязанностей и других выплат, не связанных с основной деятельностью исполнителей), и определяется по нормативу в процентах к основной заработной плате, формула (7.6).

|  |  |
| --- | --- |
|  | (7.6) |

где – дополнительная заработная плата на конкретное ПС, руб.;

– норматив дополнительной заработной платы, %.

# 6.1.2.6 Расчет отчислений в Фонд социальной защиты населения

Отчисления в Фонд социальной защиты населения (ФСЗН) определяются в соответствии с действующими законодательными актами по нормативу в процентном отношении к фонду основной и дополнительной зарплаты исполнителей, формула (7.7).

|  |  |
| --- | --- |
|  | (7.7) |

где – сумма отчислений в Фонд социальной защиты населения, руб.;

– норматив отчислений в Фонд социальной защиты населения, %.

# 

# 7.1.2.7 Расчет отчислений по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний

Отчисления по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний определяются в соответствии с действующими законодательными актами в зависимости от уровня риска отрасли, к которой относится организация-разработчик, формула (6.8).

|  |  |
| --- | --- |
|  | (7.8) |

где – сумма отчислений по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний, руб.;

– норматив отчислений по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний, %.

# 7.1.2.8 Расчет расходов на материалы

Расходы на материалы определяются с учетом действующих нормативов. По данной статье отражаются расходы на магнитную носители, перфокарты, бумагу, красящие ленты и другие материалы, необходимые для разработки ПС. Нормы расхода материалов в суммарном выражении определяются в расчете на 100 строк исходного кода. Сумма затрат материалов рассчитывается по формуле (6.9).

|  |  |
| --- | --- |
|  | (7.9) |

где – сумма расходов на материалы, руб.;

– норма расхода материалов в расчете на 100 строк исходного кода ПС, руб.

# 7.1.2.9 Расчет расходов на оплату машинного времени

Расходы включают оплату машинного времени, необходимого для разработки и отладки ПС, которое определяется по нормативам (в машино-часах) на 100 строк исходного кода машинного времени в зависимости от характера решаемых задач и типа ПЭВМ, формула (7.10).

|  |  |
| --- | --- |
|  | (7.10) |

где – сумма расходов на оплату машинного времени, руб.;

– цена одного машино-часа, руб.;

– норматив расхода машинного времени на отладку 100 строк исходного кода, машино-часов.

# 7.1.2.10 Расчет прочих прямых затрат

Расходы на конкретное ПС включают затраты на приобретение и подготовку специальной научно-технической информации и специальной литературы. Определяются по нормативу в процентах к основной заработной плате, формула (6.11).

|  |  |
| --- | --- |
|  | (7.11) |

где – сумма прочих затрат, руб.;

– норматив прочих затрат в целом по организации, %.

# 7.1.2.11 Расчет накладных расходов

Данные затраты, связанные с необходимостью содержания аппарата управления, а также с расходами на общехозяйственные нужды, относятся на конкретное ПС по нормативу в процентном отношении к основной заработной плате исполнителей, формула (6.12).

|  |  |
| --- | --- |
|  | (7.12) |

где – сумма накладных расходов, руб.;

– норматив накладных расходов в целом по организации, %.

# 7.1.2.12 Расчет суммы расходов на разработку ПС ВТ

Общая сумма расходов на ПС рассчитывается по формуле (6.13).

|  |  |
| --- | --- |
|  | (7.13) |

где – сумма расходов на разработку ПС ВТ, руб.

# 7.1.2.13 Расчет расходов на сопровождение и адаптацию

Кроме того, организация-разработчик осуществляет затраты на сопровождение и адаптацию ПС, которые определяются по нормативу, формула (6.14).

|  |  |
| --- | --- |
|  | (7.14) |

где – сумма расходов на сопровождение и адаптацию ПС ВТ, руб.;

– норматив расходов на сопровождение и адаптацию, %.

# 7.1.2.14 Расчет полной себестоимости ПС ВТ

Общая сумма расходов на разработку (с затратами на сопровождение и адаптацию) – полная себестоимость ПС определяется по формуле (6.15).

|  |  |
| --- | --- |
|  | (6.15) |

где – полная себестоимость ПС ВТ, руб.

# 7.1.2.15 Определение отпускной цены ПС ВТ

Отпускная цена определяется на основании цены разработчика, которая формируется на основе показателя рентабельности продукции. Рентабельность и прибыль по создаваемому ПС определяются исходя из результатов анализа рыночных условий, переговоров с заказчиком (потребителем) и согласования с ним отпускной цены, включающей дополнительно налог на добавленную стоимость и отчисления в целевые бюджетные фонды из выручки от реализации продукции. Прибыль рассчитывается по формуле (6.16)

|  |  |
| --- | --- |
|  | (7.16) |

где – прибыль от реализации ПС, руб.;

– уровень рентабельности ПС, %.

Прогнозируемая цена разработчика ПС без налогов, формула (6.17).

|  |  |
| --- | --- |
|  | (7.17) |

где – прогнозируемая цена разработчика ПС, руб.

Сумма налога на добавленную стоимость, формула (7.18).

|  |  |
| --- | --- |
|  | (7.18) |

где НДС – сумма налога на добавленную стоимость, руб.;

– ставка НДС, %.

Прогнозируемая отпускная цена, формула (6.19).

|  |  |
| --- | --- |
|  | (7.19) |

где – прогнозируемая отпускная цена, руб.

# 

# 7.1.3 Расчет экономического эффекта от применения ПС у пользователя

Создаваемые ПС могут предназначаться как для совершенно новых, ранее не решавшихся или решавшихся ручным способом задач, так и для традиционных задач, решаемых с помощью программных средств, которые можно совершенствовать. В результате применения нового ПС пользователь может понести значительные капитальные затраты на приобретение и освоение ПС, доукомплектования ЭВМ новыми техническими средствами и пополнение оборотных средств. Однако, если приобретенное ПС будет в достаточной степени эффективнее базового, то дополнительные капитальные затраты быстро окупятся. Эффект может быть достигнут за счет сокращения объема ПС, снижения трудоемкости подготовки данных, обработки информации, анализа результатов, уменьшения расходов машинного времени и материалов.

Для определения экономического эффекта от использования нового ПС у потребителя необходимо сравнить расходы по всем основным статьям затрат на эксплуатацию нового ПС (расходы на заработную плату с начислениями, материалы, машинное время) с расходами по соответствующим статьям базового варианта. При этом за базовый вариант следует принимать аналогичное программное средство, используемое в действующей автоматизированной системе. При сравнении базового и нового вариантов ПС в качестве экономического эффекта будет выступать общая экономия всех видов ресурсов относительно базового варианта. При этом создание нового ПС окажется экономически целесообразным лишь в том случае, если все капитальные затраты окупятся за счет получаемой экономии в ближайшие 1-2 года.

# 7.1.3.1 Исходные данные для расчета

Исходные данные для сравнения по базовому и новому вариантам приведены в таблице 7.3.

Таблица 7.3 – Исходные данные для сравнения по базовому и новому вариантам.

| Наименование показателей | Обозначения | Единицы измерения | Значение показателя | | Наименование источника информации |
| --- | --- | --- | --- | --- | --- |
| в базовом варианте | в новом варианте |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Средняя тру-доемкость ра-бот в расчете на 100 КБ |  | человеко-час на 100 КБ | 3,28 | 1,73 | По данным пользователя |
| Средний рас-ход машинного времени в рас-чете на 100 КБ |  | машино-час на 100 КБ | 6,97 | 6 | По данным пользователя |
| Средний рас-ход материалов в расчете на 100 КБ |  | руб. на 100 КБ | 0,03 | 0,02 | По данным пользователя |

Объем работ в зависимости от функциональной группы и назначения ПС определяется по формуле (7.20).

|  |  |
| --- | --- |
|  | (7.20) |

где – скорректированный объем ПС, условных машино-команд;

– коэффициент применения ПС, ед.

(машино-команд)

# 7.1.3.2 Расчет капитальных затрат

Общие капитальные вложения заказчика (потребителя), связанные с приобретением, внедрением и использованием ПС, рассчитываются по формуле (6.21).

|  |  |
| --- | --- |
|  | (7.21) |

где – затраты пользователя на приобретение ПС по отпускной цене разработчика с учетом стоимости услуг по эксплуатации и сопровождению, руб.;

– затраты пользователя на освоение ПС, руб.;

– затраты на доукомплектацию ВТ техническими средствами в связи с внедрением нового ПС, руб.;

– затраты на пополнение оборотных средств в связи с использованием нового ПС, руб.

(руб.)

Затраты на освоение ПС и на пополнение оборотных средств рассчитываются по формуле (7.22) и формуле (6.23).

|  |  |
| --- | --- |
|  | (7.22) |

(руб.)

|  |  |
| --- | --- |
|  | (7.23) |

(руб.)

# 7.1.3.3 Расчет экономики основных ресурсов в связи с использованием нового ПС

Экономия затрат на заработную плату при использовании нового ПС в расчете на объем выполненных работ, формула (6.24).

|  |  |
| --- | --- |
|  | (7.24) |

где – экономия затрат на заработную плату при решении задач с использованием нового ПС, руб.;

– экономия затрат на заработную плату при решении задач с использованием нового ПС в расчете на 100 КБ, руб.;

– объем выполненных работ с использованием нового ПС, 100 КБ.

(руб.)

Экономия затрат на заработную плату в расчете на 100 КБ, формула (6.25).

|  |  |
| --- | --- |
|  | (7.25) |

где – среднемесячная заработная плата одного программиста, руб.;

– трудоемкость работ в расчете на 100 строк кода при базовом и новом варианте соответственно, человеко-часов;

– фонд рабочего времени за месяц, ч.

При определении трудоемкости, связанной с использованием программы рекомендуется ориентироваться на показатель равный 30-50% от трудоемкости разработки в часах, формула (6.26).

|  |  |
| --- | --- |
|  | (7.26) |

Экономия начислений на заработную плату при использовании нового ПС в расчете на объем выполненных работ, формула (6.27).

|  |  |
| --- | --- |
|  | (7.27) |

где – экономия начислений на заработную плату при решении задач с использованием нового ПС, руб.;

– коэффициент начислений на заработную плату, ед.

|  |  |
| --- | --- |
|  | (7.28) |

Экономия затрат на оплату машинного времени в расчете на выполненный объем работ в результате применения нового ПС, формула (7.29).

|  |  |
| --- | --- |
|  | (7.29) |

где – экономия затрат на оплату машинного времени при решении задач с использованием нового ПС, руб.;

– экономия затрат на оплату машинного времени при решении задач с использованием нового ПС в расчете на 100 КБ, руб.

Экономия затрат на оплату машинного времени в расчете на 100 КБ, формула (7.30).

|  |  |
| --- | --- |
|  | (7.30) |

где – средний расход машинного времени в расчете на 100 КБ при применении базового и нового варианта ПС соответственно, машино-часов.

Экономия затрат на материалы при использовании нового ПС в расчете на объем выполненных работ, формула (7.31).

|  |  |
| --- | --- |
|  | (7.31) |

где – экономия затрат на материалы при использовании нового ПС, руб.;

– экономия затрат на материалы в расчете на 100 КБ при использовании нового ПС, руб.

|  |  |
| --- | --- |
|  | (7.32) |

где – средний расход материалов у пользователя в расчете на 100 КБ при использовании базового и нового варианта ПС соответственно, руб.

Общая годовая экономия текущих затрат, связанных с использованием нового ПС, формула (7.33).

|  |  |
| --- | --- |
|  | (7.33) |

# 7.1.3.4 Расчет экономического эффекта

# 

Внедрение нового ПС позволит пользователю сэкономить на текущих затратах, то есть практически получить на эту сумму дополнительную прибыль. Для пользователя в качестве экономического эффекта выступает лишь чистая прибыль – дополнительная прибыль, остающаяся в его распоряжении, формула (6.34).

|  |  |
| --- | --- |
|  | (7.34) |

где – прирост прибыли, руб.;

– ставка налога на прибыль, %.

|  |  |
| --- | --- |
|  | (7.35) |

В процессе использования нового ПС чистая прибыль в конечном итоге возмещает капитальные затраты. Однако, полученные при этом суммы результатов (прибыли) и затрат (капитальных вложений) по годам приводят к единому времени – расчетному году (за расчетный год принят год разработки ДП) путем умножения результатов и затрат за каждый год на коэффициент привидения (), который рассчитывается по формуле (6.36).

|  |  |
| --- | --- |
|  | (7.36) |

где – норматив привидения разновременных затрат и результатов;

– расчетный год, ;

– номер года, результаты и затраты которого приводятся к расчетному.

Данные расчета экономического эффекта сведены в таблицу 7.4.

Таблица 7.4 – Расчёт экономического эффекта от использования нового ПС

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Показатель | Единица измерения | | Методичка расчёта | Год | | | | | |
| 2020 | | 2021 | 2022 | 2023 | 2024 |
| Результаты: | | | | | | | | | |
| Прирост при-были за счёт экономии затрат | руб. |  | | |  |  |  |  |  |
| Сумма прибыли с учётом фактора времени | руб. |  | | |  | 1042,72 | 962,51 | 882,30 | 802,09 |
| Затраты: | | | | | | | | | |
| Затраты на при-обретение ПС | руб. |  | | | 2390,4 | – | – | – | – |
| Затраты на освоение ПС | руб. |  | | | 23,90 | – | – | – | – |
| Затраты на по-полнение обо-ротных средств | руб. |  | | | 47,80 | 47,80 | 47,80 | 47,80 | 47,80 |
| Сумма затрат | руб. |  | | | 2462,11 | 47,80 | 47,80 | 47,80 | 47,80 |
| Экономи-ческий эффект | руб. |  | | | -1316,26 | 999,23 | 922,36 | 845,50 | 768,63 |
| Экономический эффект с нараста-ющим итогом | руб. |  | | | -1316,26 | -317,03 | 605,33 | 1450,83 | 2219,46 |

# 

# 7.1.4 Вывод

Внедрение новых изделий имеет первостепенное значение для дальнейшего развития предприятия. В результате расчетов было найдено наиболее рациональное программное решение, обеспечивающее высокий технический уровень программы и дающее существенную экономию ресурсов, как при разработке проекта, так и при его реализации у пользователя (покупателя, заказчика).

Из проведенных расчетов было выявлено, что прогнозируемая отпускная цена разработанного программного средства составляет 2390 белорусских рублей. Рассчитанные затраты полностью окупаются и оправдывают стоимость программного средства.

Помимо этого, был рассчитан экономический эффект от использования программного средства у конечного пользователя на протяжении четырех лет, начиная с 2020 и заканчивая 2023 годом:

* в первый (2020) год – -1316 белорусских рублей;
* во второй (2021) год – 999 белорусских рублей;
* в третий (2022) год – 922 белорусских рублей;
* в четвертый (2023) год – 845 белорусских рублей;
* в пятый (2024) год – 768 белорусских рублей.

На основании расчетов можно сделать вывод, что экономический эффект от использования данного программного средства в несколько раз превысит затраты, и окупит затраты через два года.

Исходя из вышеизложенного следует, целесообразность использования программного средства социальной сети с возможностью передачи файлов «TalkIn».

Вывод

В ходе проделанной работы был создан интернет-магазин радиоуправляемой технологий с функциями просмотра каталогов и заказа товаров. Был реализован дизайн сайта, который является простым и легким в понимании для любого пользователя. На сайте используется база данных, которой может управлять администратор. Также, на сайте имеются менеджеры, которые имеют доступ к добавлению товаров в базу данных, обслуживанию заказов.

Все разработанные функции являются обязательным для любого интернет-магазина, для удовлетворения желаний клиента и для достойной конкуренции.

В ходе работы были достигнуты следующие цели:

* разработана схема базы данных;
* реализована схема базы данных;
* разработан дизайн интернет-магазина;
* созданы формы для ввода данных;
* автоматизирована работа созданной базы данных;
* изучены особенности процесса работы продавца.

Таким образом, в результате проделанной работы был создан интернет-магазин мягкой мебели.

Исходя из проведенных выше расчетов, в первый же год использования ПС приведет к прибыли. В последующие года программный продукт начнет приносить всё больше прибыль, так как экономический эффект равен 536,79. Можно сделать вывод, что экономический эффект за каждый год является положительным, более того экономический эффект с нарастающим итогом с каждым годом становится всё больше и больше. Это говорит о том, что разработка данного программного средства является выгодной, затраты на разработку окупятся в первый год, и в последующих годах программное средство становится выгодным.

Заключение

В рамках дипломного проекта было разработана «Веб-приложение по продаже радиоуправляемых технологий», предназначенная для продажи товаров в сети интернет и расширения клиентской базы предприятия.

В отличие прочих сайтов, приложений, проектируемая автоматизированная система должна обеспечить простую навигацию по списку всех доступный функций, быстрый доступ в любую точку сайта.

В автоматизированной системе продаж строительных материалов присутствуют такие функции, как:

* фильтрация товаров по категориям — для предоставления пользователю возможности быстрого перехода к нужной категории товара;
* удобная карточка товара — содержит фото, название, изображение, описание, а также стоимость товара;
* система обновления товаров ­ — администратор всегда может быстро обновить информацию о любом товаре, либо добавить новый товар или категорию;
* управление поступающими заказами — после оформления заказа покупателем, данные с пользовательских форм поступают к администратору сайта, уведомляя его о поступившем заказе;
* управление акциями и скидками — цену и описание всегда можно изменить в панели администратора в несколько нажатий клавиш;
* корзина товаров — очень важная составляющая любого интернет-магазина, так как с её помощью пользователь может выводить несколько различных либо одинаковых товаров одним заказом;
* быстрая покупка в несколько кликов — функция, направленная на удобство использования программного средства покупателем;
* управление товарами интернет-магазина с помощью панели администратора — функция управления сайтом администратором.

В разделе «Охрана труда» были рассмотрены вопросы трудоохранного анализа информационной технологии и её оздоровление, описываются особенности умственного труда пользователя при использовании созданной технологии. Определены влияния элементов рабочего места в организации труда на работоспособность и здоровье операторов, и описан рациональный режима труда и отдыха для снижения умственного утомления.

В экономическом разделе были произведены расчеты основных экономических показателей (себестоимость, затраты и дополнительные величины). После расчета экономического эффекта на 4 года, было выявлено, что приложение окупает затраты на его разработку.

Список использованных источников

Литература

1. Акила, К. Интернет магазин с нуля / К. Акила, А. Еремеевский. – Санкт-Петербург, 2013. – 176 с.

### Викрам, В. Zend Framework. Разработка веб-приложений на PHP / В. Викрам. - [Санкт-Петербург](https://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%BD%D0%BA%D1%82-%D0%9F%D0%B5%D1%82%D0%B5%D1%80%D0%B1%D1%83%D1%80%D0%B3), 2012. – 244 с.

1. Дари, К. PHP и MySQL: создание интернет магазина / К. Дари, Э. Баланеску. - М.: Вильямс, 2016 – 640 с.
2. Зандстра, М. PHP. Объекты, шаблоны и методики программирования, 2-е издание / М. Зандстра. – М.: Вильямс, 2010. – 541 с.
3. Ильин, В. Моделирование бизнес-процессов. Практический опыт разработчика / В. Ильин. – Мн.: САЛВИЯ, 2016 – 252 с.
4. [Кормен, Т. Алгоритмы. Построение и анализ](http://vk.com/wall-54530371_2325) / Т. Кормен, Ч. Лейзерсон, Р. Ривест . – М.: Форум, ИНФРА, 2013. – 588 c.
5. Линн, Б. Изучаем PHP и MySQL / Б. Линн, М. Моррисон. – М. Эксмо, 2010. – 682 с.
6. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript / Р. Никсон. - [Санкт-Петербург](https://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%BD%D0%BA%D1%82-%D0%9F%D0%B5%D1%82%D0%B5%D1%80%D0%B1%D1%83%D1%80%D0%B3), 2011. – 237 с.

**Нормативные документы**

1. ГОСТ 12.1.001-89 ССБТ. Ультразвук. Общие требования безопасности
2. ГОСТ 12.1.029-80 ССБТ. Средства и методы защиты от шума. Классификация.
3. Шатило, С. Н. Исследование производственного шума : учебно-методическое пособие / С. Н. Шатило, С. В. Дорошко, В. В. Карпенко. – Гомель : БелГУТ, 2011. – 61 с.
4. Трудовой кодекс Республики Беларусь. – Мн. : Амалфея, 2015. – 316 с.

Приложение А

(обязательное)

Текст программы

Файл index.php

<?php

define('myeshop', true);

require\_once('include/Exceptions.php');

require\_once('utility/pager.php');

include "include/DB.php";

include ('utility/handleData.php');

include ('utility/groupPrice.php');

session\_start();

include "include/auth-cookie.php";

try {

// получаем значение для сортировки товара

if (isset ($\_GET['sort'])) {

$sorting = $\_GET['sort'];

} else {

$sorting = '';

}

switch ($sorting) {

case 'price-asc':

$sorting = 'price ASC';

$sort\_name = 'От дешевых к дорогим';

break;

case 'price-desc':

$sorting = 'price DESC';

$sort\_name = 'От дорогих к дешевым';

break;

case 'popular':

$sorting = 'count DESC';

$sort\_name = 'Популярные';

break;

case 'news':

$sorting = 'datetime DESC';

$sort\_name = 'Новинки';

break;

case 'brand':

$sorting = 'brand ASC';

$sort\_name = 'По алфавиту';

break;

default:

$sorting = 'products\_id ASC';

$sort\_name = 'Нет сортировки';

break;

}

?>

<!DOCTYPE html>

<html>

<head>

<title>Shop</title>

<meta http-equiv="content-type" content="text/html; charset=UTF-8" />

<link href="css/reset.css" rel="stylesheet" type="text/css" />

<link href="css/style.css" rel="stylesheet" type="text/css" />

<link href="TrackBar/jQuery/trackbar.css" rel="stylesheet" type="text/css" />

<script type="text/javascript" src="js/jquery-2.1.1.js"></script>

<script type="text/javascript" src="js/jcarouserllite\_1.0.1.js"></script>

<script type="text/javascript" src="js/jquery.cookie.js"></script>

<script type="text/javascript" src="TrackBar/jQuery/jquery.trackbar.js"></script>

<script type="text/javascript" src="js/jquery.TextChange.js"></script>

<script type="text/javascript" src="js/shop-script.js"></script>

</head>

<body>

<div id="block-body">

<!-- подключаем блок block-header-->

<?php include('include/block-header.php'); ?>

<div id="block-right">

<!-- подключаем блок block-category-->

<?php include('include/block-category.php'); ?>

<!-- подключаем блок block-parameter-->

<?php include('include/block-parameter.php'); ?>

<!-- подключаем блок block-news-->

<?php include('include/block-news.php'); ?>

</div>

<div id="block-content">

<div id="block-sorting">

<p id="nav-breadcrumbs"><a href="index.php">Главная страница</a> \ <span>Все товары</span></p>

<ul id="options-list">

<li>Вид: </li>

<li><img id="style-grid" src="images/icon-grid.png" /></li>

<li><img id="style-list" src="images/icon-list.png" /></li>

<li>Сортировать: </li>

<li><a href="#" id="select-sort"><?php echo $sort\_name; ?> </a>

<ul id="sorting-list">

<li><a href="index.php?sort=price-asc" >От дешевых к дорогим</a> </li>

<li><a href="index.php?sort=price-desc" >От дорогих к дешевым</a> </li>

<li><a href="index.php?sort=popular" >Популярное</a> </li>

<li><a href="index.php?sort=news" >Новинки</a> </li>

<li><a href="index.php?sort=brand" >От А до Я</a> </li>

</ul>

</li>

</ul>

</div>

<ul id="block-tovar-grid">

<?php

if (isset( $\_GET['sort'])) {

$sortPar = '&sort='.$\_GET['sort'];

} else {

$sortPar = '';

}

$parameters = $sortPar;

/\*\*

\* Постраничная навигация

\*/

$pnumber = 10; // количество выводимых предметов

$pageLink = 5; // количество ссылок слева и справа

if( isset($\_GET['page'])) {

$page = intval($\_GET['page']); // номер страницы

} else {

$page = 0;

}

$page = intval($page);

$sth = DB::getStatement("SELECT COUNT(\*) as count FROM table\_products WHERE visible=?");

$sth->execute(array(1));

$rows = $sth->fetch();

$total = $rows['count']; // всего позиций

$number = $total / $pnumber; // количество ссылок на странице

if ($total / $pnumber != 0 ) $number++; // если не равно 0, то добавляем 1

$number = intval($number); // и приводим к целому числу

if (empty($page) || $page < 0 ) $page = 1; // номер страницы

if ($page > $number) $page = $number; // если страница больше общего числа, то она и есть максимальная

$start = $page \* $pnumber - $pnumber; // с какого id выводить товар

$queryStart = " LIMIT $start, $pnumber";

$sth = DB::getStatement("SELECT \* FROM table\_products WHERE visible=? ORDER BY $sorting $queryStart");

$sth->execute(array(1));

$rows = $sth->fetchAll();

// echo "<tt><pre> - asjfksajdfklsdjf - ".print\_r(gettype($pnumber), true). "</pre></tt>";

foreach ($rows as $row):

if (isset($row['image']) && file\_exists('uploads\_images/'.$row['image'])) {

$img\_path = 'uploads\_images/'.$row['image'];

$max\_width = 200;

$max\_height = 200;

list($width, $height) = getimagesize($img\_path);

$ratioh = $max\_height / $height;

$ratiow = $max\_width / $width;

$ratio = min($ratioh, $ratiow);

$width = intval($ratio \* $width);

$height = intval($ratio \* $height);

} else {

$img\_path = "images/no-image.png";

$width = 110;

$height = 200;

}

// количество отзывов

$sthReview = DB::getStatement('SELECT COUNT(\*) as count FROM table\_reviews WHERE products\_id = ? AND moderat = ?');

$sthReview->execute(array($row['products\_id'], 1));

$rowReview = $sthReview->fetch();

if ($rowReview['count'] >=0 ) {

$count\_review = $rowReview['count'];

}

?>

<li>

<div class="block-images-grid"><img src="<?php echo $img\_path; ?>" width="<?php echo $width; ?>" height="<?php echo $height; ?>" /></div>

<p class="style-title-grid"><a href="view\_content.php?id=<?php echo $row['products\_id']; ?>" ><?php echo $row['title']; ?></a></p>

<ul class="reviews-and-counts-grid">

<li><img src="images/eye-icon.png" /><p><?php echo $row['count']; ?></p></li>

<li><img src="images/comment-icon.png" /><p><?php echo $count\_review; ?></p></li>

</ul>

<a href="" class="add-cart-style-grid" tid="<?php echo $row['products\_id']; ?>"></a>

<p class="style-price-grid"><strong><?php echo groupPrice($row['price']); ?></strong> руб.</p>

<div class="mini-features"><?php echo $row['mini\_features']; ?></div>

</li>

<?php endforeach; ?>

</ul>

<ul id="block-tovar-list">

<?php

// $sth = DB::getStatement("SELECT \* FROM table\_products WHERE visible='1' ORDER BY {$sorting}");

// $sth->execute();

// $rows = $sth->fetchAll();

foreach ($rows as $row):

if (isset($row['image']) && file\_exists('uploads\_images/'.$row['image'])) {

$img\_path = 'uploads\_images/'.$row['image'];

$max\_width = 150;

$max\_height = 150;

list($width, $height) = getimagesize($img\_path);

$ratioh = $max\_height / $height;

$ratiow = $max\_width / $width;

$ratio = min($ratioh, $ratiow);

$width = intval($ratio \* $width);

$height = intval($ratio \* $height);

} else {

$img\_path = "images/noimages80x70.png";

$width = 80;

$height = 70;

}

// количество отзывов

$sthReview = DB::getStatement('SELECT COUNT(\*) as count FROM table\_reviews WHERE products\_id = ? AND moderat = ?');

$sthReview->execute(array($row['products\_id'], 1));

$rowReview = $sthReview->fetch();

if ($rowReview['count'] >=0 ) {

$count\_review = $rowReview['count'];

}

//echo "<tt><pre>".print\_r($row['image'], true). "</pre></tt>";

?>

<li>

<div class="block-images-list"><img src="<?php echo $img\_path; ?>" width="<?php echo $width; ?>" height="<?php echo $height; ?>" /></div>

<ul class="reviews-and-counts-list">

<li><img src="images/eye-icon.png" /><p><?php echo $row['count']; ?></p></li>

<li><img src="images/comment-icon.png" /><p><?php echo $count\_review; ?></p></li>

</ul>

<p class="style-title-list"><a href="view\_content.php?id=<?php echo $row['products\_id']; ?>"" ><?php echo $row['title']; ?></a></p>

<a href="" class="add-cart-style-list" tid="<?php echo $row['products\_id']; ?>"></a>

<p class="style-price-list"><strong><?php echo groupPrice($row['price']); ?></strong> руб.</p>

<div class="style-text-list"><?php echo $row['mini\_description']; ?></div>

</li>

<?php endforeach; ?>

</ul>

<?php

echo '<div class="pstrnav"><ul>';

echo pager( $page, $pageLink, $number, $total, $parameters);

echo '</ul></div>';

?>

</div><!-- end block-content -->

<!-- подключаем блок block-random-->

<?php include('include/block-random.php'); ?>

<!-- подключаем блок block-footer-->

<?php include('include/block-footer.php'); ?>

</div>

</body>

</html>

<?php

} catch(PDOException $ex) {

throw new Exceptions($ex);

}

?>

Файл cart.php

<?php

define('myeshop', true);

require\_once('include/Exceptions.php');

require\_once('utility/pager.php');

include "include/DB.php";

include ('utility/handleData.php');

include ('utility/getIP.php');

include ('utility/groupPrice.php');

session\_start();

include "include/auth-cookie.php";

if (isset($\_GET['id'])) {

$id = handleData($\_GET['id']);

}

if (isset($\_GET['action'])) {

$action = handleData($\_GET['action']);

}

$ip = getIP();

switch ($action) {

case 'clear':

$sth = DB::getStatement("DELETE FROM cart WHERE cart\_ip = ?");

$sth->execute(array($ip));

break;

case 'delete':

$sth = DB::getStatement("DELETE FROM cart WHERE cart\_id = ? AND cart\_ip = ?");

$sth->execute(array($id, $ip));

break;

}

if (isset($\_POST["submitdata"]) || isset($\_POST['pay now'])) {

$sth\_orders\_insert = DB::getStatement('INSERT INTO orders (order\_datetime,order\_dostavka,

order\_fio,order\_address,order\_phone,

order\_note,order\_email)

VALUES (?,?,?,?,?,?,?)');

$date = date('Y-m-d H:i:s', time());

if ($\_SESSION['auth'] == 'yes\_auth'){

$sth\_orders\_insert->execute(array($date,

$\_POST['order\_delivery'],

"{$\_SESSION['auth\_surname']} {$\_SESSION['auth\_name']} {$\_SESSION['auth\_patronymic']}",

$\_SESSION['auth\_address'],

$\_SESSION['auth\_phone'],

$\_POST['order\_note'],

$\_SESSION['auth\_email']));

$\_SESSION['order\_delivery'] = $\_POST['order\_delivery'];

$\_SESSION['order\_payment'] = $\_POST['order\_payment'];

$\_SESSION['order\_note'] = $\_POST['order\_note'];

$\_SESSION['order\_id'] = DB::getId();

} else {

$\_SESSION['order\_delivery'] = $\_POST['order\_delivery'];

$\_SESSION['order\_payment'] = $\_POST['order\_payment'];

$\_SESSION['order\_fio'] = $\_POST['order\_fio'];

$\_SESSION['order\_email'] = handleData($\_POST['order\_email']);

$\_SESSION['order\_phone'] = handleData($\_POST['order\_phone']);

$\_SESSION['order\_address'] = $\_POST['order\_address'];

$\_SESSION['order\_note'] = $\_POST['order\_note'];

$sth\_orders\_insert->execute(array($date,

$\_POST['order\_delivery'],

$\_POST['order\_fio'],

$\_POST['order\_address'],

$\_POST['order\_phone'],

$\_POST['order\_note'],

$\_POST['order\_email']));

$\_SESSION['order\_id'] = DB::getId();

}

$sth\_cart\_select = DB::getStatement('SELECT \* FROM cart WHERE cart\_ip=?');

$sth\_cart\_select->execute(array($ip));

$rows\_cart\_select = $sth\_cart\_select->fetchAll();

if (!empty($rows\_cart\_select)) {

foreach ($rows\_cart\_select as $row\_cart\_select) {

$sth\_products\_insert = DB::getStatement('INSERT INTO buy\_products (buy\_id\_order,

buy\_id\_product,

buy\_count\_product)

VALUES (?,?,?)');

$sth\_products\_insert->execute(array($\_SESSION['order\_id'],

$row\_cart\_select['cart\_id\_product'],

$row\_cart\_select['cart\_count']));

}

}

header("Location: cart.php?action=completion");

}

$sth1 = DB::getStatement("SELECT \* FROM cart, table\_products WHERE cart.cart\_ip = '{$ip}' AND table\_products.products\_id = cart.cart\_id\_product");

$sth1->execute();

$rows1 = $sth1->fetchAll();

if (! empty($rows1)) {

$itogPriceCart = 0;

$int1 = 0;

foreach ($rows1 as $row1) {

$int1 = $int1 + ($row1["price"] \* $row1["cart\_count"]);

}

$itogPriceCart = groupPrice($int1);

$\_SESSION['itog\_price'] = $int1;

}

try {

?>

<!DOCTYPE html>

<html>

<head>

<title>Корзина заказов</title>

<meta http-equiv="content-type" content="text/html; charset=UTF-8" />

<link href="css/reset.css" rel="stylesheet" type="text/css" />

<link href="css/style.css" rel="stylesheet" type="text/css" />

<link href="TrackBar/jQuery/trackbar.css" rel="stylesheet" type="text/css" />

<script type="text/javascript" src="js/jquery-2.1.1.js"></script>

<script type="text/javascript" src="js/jcarouserllite\_1.0.1.js"></script>

<script type="text/javascript" src="js/jquery.cookie.js"></script>

<script type="text/javascript" src="TrackBar/jQuery/jquery.trackbar.js"></script>

<script type="text/javascript" src="js/jquery.TextChange.js"></script>

<script type="text/javascript" src="js/shop-script.js"></script>

</head>

<body>

<div id="block-body">

<?php include('include/block-header.php'); ?>

<div id="block-right">

<?php include('include/block-category.php'); ?>

<?php include('include/block-parameter.php'); ?>

<?php include('include/block-news.php'); ?>

</div>

<div id="block-content">

<?php

$action = handleData($\_GET['action']);

switch ($action) {

case 'oneclick':

// блок навигации

echo '<div id="block-step">

<div id="name-step">

<ul>

<li><a class="active" href="" >1. Корзина товаров</a></li>

<li><span>&rarr;</span></li>

<li><a href="" >2. Контактная информация</a></li>

<li><span>&rarr;</span></li>

<li><a href="" >3. Завершение</a></li>

</ul>

</div>

<p>Шаг 1 из 3</p>

<a href="cart.php?action=clear">Очистить</a>

</div>';

// заголовок таблицы блока

echo '<div id="header-list-cart">

<div id="head1">Изображение</div>

<div id="head2">Наименование товара</div>

<div id="head3">Кол-во</div>

<div id="head4">Цена</div>

</div>';

$sth = DB::getStatement("SELECT \* FROM cart, table\_products WHERE cart.cart\_ip = '{$ip}' AND table\_products.products\_id = cart.cart\_id\_product");

$sth->execute();

$rows = $sth->fetchAll();

if (! empty($rows)) {

$allPrice = 0;

foreach ($rows as $row) {

$int = $row['cart\_price'] \* $row['cart\_count'];

$allPrice = $allPrice + $int;

if (strlen($row['image']) > 0 && file\_exists('uploads\_images/'.$row['image'])) {

$imgPath = 'uploads\_images/'.$row['image'];

$maxWidth = 100;

$maxHeight = 100;

list($width, $height) = getimagesize($imgPath);

$ratioh = $maxHeight / $height;

$ratiow = $maxWidth / $width;

$ratio = min($ratioh, $ratiow);

$width = intval($ratio \* $width);

$height = intval($ratio \* $height);

} else {

$imgPath = 'images/noimages.jpeg';

$width = 120;

$height = 105;

}

echo '<div class="block-list-cart">

<div class="img-cart">

<p align="center"><img src="'.$imgPath.'" width="'.$width.'" height="'.$height.'" /></p>

</div>

<div class="title-cart">

<p><a href="view\_content.php?id='.$row["products\_id"].'" >'.$row["title"].'</a></p>

<p class="cart-mini-features">'.$row["mini\_features"].'</p>

</div>

<div class="count-cart">

<ul class="input-count-style">

<li><p align="center" class="count-minus" iid="'.$row['cart\_id'].'">-</p></li>

<li><p align="center"><input id="input-id'.$row['cart\_id'].'" iid="'.$row['cart\_id'].'" class="count-input" maxlength="3" type="text" value="'.$row["cart\_count"].'" /></p></li>

<li><p align="center" class="count-plus" iid="'.$row['cart\_id'].'">+</p></li>

</ul>

</div>

<div id="tovar'.$row['cart\_id'].'" class="price-product"><h5><span class="span-count">'.$row["cart\_count"].'</span> x <span>'.groupPrice($row["cart\_price"]).' руб</span></h5><p price="'.$row['cart\_price'].'">'.groupPrice($int).' руб</p></div>

<div class="delete-cart"><a href="cart.php?id='.$row["cart\_id"].'&action=delete" ><img src="images/bsk\_item\_del.png" /></a></div>

<div id="bottom-cart-line"></div>

</div>';

}

echo '<h2 class="itog-price" align="right">Итого: <strong>'.groupPrice($allPrice).'</strong> руб</h2>

<p align="right" class="button-next"><a href="cart.php?action=confirm">Далее</a></p>';

} else {

echo '<h3 id="clear-cart" align="center">Корзина пуста</h3>';

}

break;

// переход к оплате

case 'confirm':

echo '<div id="block-step">

<div id="name-step">

<ul>

<li><a href="cart.php?action=oneclick">1. Корзина товаров</a></li>

<li><span>&rarr;</span></li>

<li><a class="active" href="cart.php?action=confirm">2. Контактная информация</a></li>

<li><span>&rarr;</span></li>

<li><a href="" >3. Завершение</a></li>

</ul>

</div>

<p>Шаг 2 из 3</p>

</div>';

if (isset($\_SESSION['order\_delivery']) && $\_SESSION['order\_delivery'] != '') {

$orderDelivery = $\_SESSION['order\_delivery'];

} else {

$orderDelivery = $\_POST['order\_delivery'];

}

$chck1 = $chck2 = $chck3 = '';

if ($orderDelivery == "По почте") $chck1 = "checked";

if ($orderDelivery == "Курьером") $chck2 = "checked";

if ($orderDelivery == "Самовывоз") $chck3 = "checked";

if (isset($\_SESSION['order\_payment']) && $\_SESSION['order\_payment'] != '') {

$orderPayment = $\_SESSION['order\_payment'];

} else {

$orderPayment = $\_POST['order\_payment'];

}

$chk4 = $chk5 = '';

if ($orderPayment == 'Другие виды оплаты') $chk4 = "checked";

if ($orderPayment == 'PayPal') $chk4 = "checked";

echo '<h3 class="title-h3">Способы доставки: </h3>';

if (isset($\_SESSION["order\_msg"]) && $\_SESSION["order\_msg"]) {

echo $\_SESSION["order\_msg"];

unset($\_SESSION["order\_msg"]);

}

echo '<form method="POST">

<ul id="info-radio">

<li>

<input type="radio" name="order\_delivery" class="order\_delivery" id="order\_delivery1" value="По почте" ' .$chck1.' />

<label class="label\_delivery" for="order\_delivery1">По почте</label>

</li>

<li>

<input type="radio" name="order\_delivery" class="order\_delivery" id="order\_delivery2" value="Курьером" '.$chck2.' />

<label class="label\_delivery" for="order\_delivery2">Курьером</label>

</li>

<li>

<input type="radio" name="order\_delivery" class="order\_delivery" id="order\_delivery3" value="Самовывоз" ' .$chck3.' />

<label class="label\_delivery" for="order\_delivery3">Самовывоз</label>

</li>

</ul>

<h3 class="title-h3">Способ оплаты: </h3>

<ul id="info-radio">

<li>

<input type="radio" name="order\_payment" class="order\_payment" id="order\_payment1" value="Другие виды оплаты" ' .$chck1.' />

<label class="label\_payment" for="order\_payment1">Другие виды оплаты</label>

</li>

<li>

<input type="radio" name="order\_payment" class="order\_payment" id="order\_payment2" value="PayPal" '.$chck2.' />

<label class="label\_payment" for="order\_payment2">PayPal</label>

</li>

</ul>

<h3 class="title-h3">Информация для доставки: </h3>

<ul id="info-order">';

if ($\_SESSION['auth'] != 'yes\_auth') {

if (isset($\_SESSION["order\_fio"]) && $\_SESSION["order\_fio"] != '') {

$orderFio = $\_SESSION["order\_fio"];

} else {

$orderFio = $\_POST['order\_fio'];

}

if (isset($\_SESSION["order\_email"]) && $\_SESSION["order\_email"] != '') {

$orderEmail = $\_SESSION["order\_email"];

} else {

$orderEmail = $\_POST['order\_email'];

}

if (isset($\_SESSION["order\_phone"]) && $\_SESSION["order\_phone"] != '') {

$orderPhone = $\_SESSION["order\_phone"];

} else {

$orderPhone = $\_POST['order\_phone'];

}

if (isset($\_SESSION["order\_address"]) && $\_SESSION["order\_address"] != '') {

$orderAddress = $\_SESSION["order\_address"];

} else {

$orderAddress = $\_POST['order\_address'];

}

echo '<li><label for="order\_fio"><span>\*</span>ФИО</label><input type="text" name="order\_fio" id="order\_fio" value="'.$orderFio.'" /><span class="order\_span\_style">Пример: Иванов Иван Иванович</span></li>

<li><label for="order\_email"><span>\*</span>E-mail</label><input type="text" name="order\_email" id="order\_email" value="'.$orderEmail.'" /><span class="order\_span\_style">Пример: ivanov@mail.ru</span></li>

<li><label for="order\_phone"><span>\*</span>Телефон</label><input type="text" name="order\_phone" id="order\_phone" value="'.$orderPhone.'" /><span class="order\_span\_style">Пример: 8 950 333-22-11</span></li>

<li><label class="order\_label\_style" for="order\_address"><span>\*</span>Адрес<br /> доставки</label><input type="text" name="order\_address" id="order\_address" value="'.$orderAddress.'" /><span class="order\_span\_style">Пример: г. Москва,<br /> ул. Строителей 35, корп.2, кв.35 </span></li>';

}

if (isset($\_SESSION["order\_note"]) && $\_SESSION["order\_note"] != '') {

$orderNote = $\_SESSION["order\_note"];

} else {

$orderNote = $\_POST['order\_note'];

}

echo '<li><label class="order\_label\_style" for="order\_note">Примечание</label>

<textarea name="order\_note">'.$orderNote.'</textarea><span>Уточните информацию о заказе.<br /> Например, удобное время для звонка<br /> нашего менеджера</span></li></ul>

<p align="right"><input type="submit" name="submitdata" id="confirm-button-next" value="Далее" /></p>

</form>';

break;

case 'completion':

echo '<div id="block-step">

<div id="name-step">

<ul>

<li><a href="cart.php?action=oneclick">1. Корзина товаров</a></li>

<li><span>&rarr;</span></li>

<li><a href="cart.php?action=confirm">2. Контактная информация</a></li>

<li><span>&rarr;</span></li>

<li><a class="active" href="" >3. Завершение</a></li>

</ul>

</div>

<p>Шаг 3 из 3</p>

</div>

<h3>Конечная информация: </h3>';

if ($\_SESSION['auth'] == 'yes\_auth') { // если пользователь авторизован

echo '<ul id="list-info">

<li><strong>Способ доставки: </strong>'.$\_SESSION['order\_delivery'].'</li>

<li><strong>E-mail: </strong>'.$\_SESSION['auth\_email'].'</li>

<li><strong>ФИО: </strong>'.$\_SESSION['auth\_surname'].' '.$\_SESSION['auth\_name'].' '.$\_SESSION['auth\_patronymic'].'</li>

<li><strong>Адрес доставки: </strong>'.$\_SESSION['auth\_address'].'</li>

<li><strong>Телефон: </strong>'.$\_SESSION['auth\_phone'].'</li>

<li><strong>Примечание: </strong>'.$\_SESSION['order\_note'].'</li>

</ul>';

} else { // если пользователь не авторизован

echo '<ul id="list-info">

<li><strong>Способ доставки: </strong>'.$\_SESSION['order\_delivery'].'</li>

<li><strong>E-mail: </strong>'.$\_SESSION['order\_email'].'</li>

<li><strong>ФИО: </strong>'.$\_SESSION['order\_fio'].'</li>

<li><strong>Адрес доставки: </strong>'.$\_SESSION['order\_address'].'</li>

<li><strong>Телефон: </strong>'.$\_SESSION['order\_phone'].'</li>

<li><strong>Примечание: </strong>'.$\_SESSION['order\_note'].'</li>

</ul>';

}

if ($\_SESSION['order\_payment'] == 'PayPal') {

?>

<form action="https://www.sandbox.paypal.com/cgi-bin/webscr" method="post">

<input type="hidden" name="cmd" value="\_cart" >

<input type="hidden" name="upload" value="1" >

<input type="hidden" name="business" value="zhalninpal-facilitator@me.com" >

<?php

$sth = DB::getStatement("SELECT \* FROM cart, table\_products WHERE cart.cart\_ip = '{$ip}' AND table\_products.products\_id = cart.cart\_id\_product");

$sth->execute();

$rows = $sth->fetchAll();

if (! empty($rows)) {

$allPrice = 0;

$i = 1;

$sum\_subtotal = 0;

$sum\_shipping = 0;

// echo "<tt><pre>".print\_r($rows , true)."</pre></tt>";

foreach ($rows as $row) {

$int = $row['cart\_price'] \* $row['cart\_count'];

$allPrice = $allPrice + $int;

?>

<input type="hidden" name="item\_name\_<?php echo $i; ?>" value="<?php echo $row['title']; ?>" >

<input type="hidden" name="item\_number\_<?php echo $i; ?>" value="<?php echo $\_SESSION['order\_id']; ?>" >

<input type="hidden" name="amount\_<?php echo $i; ?>" value="<?php echo $row['price']; ?>" >

<input type="hidden" name="quantity\_<?php echo $i; ?>" value="<?php echo $row['cart\_count']; ?>" >

<?php

$i++;

}

if( $allPrice <= 10 ) {

$sum\_shipping = $allPrice;

} else {

$sum\_shipping = round($allPrice / 100 \* 4.02 + 10);

}

}

?>

<input type="hidden" name="currency\_code" value="RUB" >

<input type="hidden" name="lc" value="RUS" >

<input type="hidden" name="rm" value="2" >

<input type="hidden" name="shipping\_1" value="<?php echo $sum\_shipping; ?>" >

<input type="hidden" name="return" value="http://zhalnin.tmweb.ru/" >

<input type="hidden" name="cancel\_return" value="http://zhalnin.tmweb.ru/" >

<input type="hidden" name="notify\_url" value="http://zhalnin.tmweb.ru/order/paypal.php" >

<input type="image" src="images/paypal/paypal\_mini.png" name="pay now" value="pay" class="pay-button" width="150" height="37" />

</form>

<?php

} else {

echo '<h2 class="itog-price" align="right">Итого: <strong>'.$itogPriceCart.'</strong> руб</h2>

<form method="post" action="https://www.walletone.com/checkout/default.aspx" accept-charset="UTF-8">

<input type="hidden" name="WMI\_MERCHANT\_ID" value="169513168489"/>

<input type="hidden" name="WMI\_PAYMENT\_AMOUNT" value="0"/>

<input type="hidden" name="WMI\_CURRENCY\_ID" value="643"/>

<input type="hidden" name="WMI\_PAYMENT\_NO" value="'.$\_SESSION["order\_id"].'"/>

<input type="hidden" name="WMI\_DESCRIPTION" value="Оплата демонстрационного заказа"/>

<input type="hidden" name="WMI\_SUCCESS\_URL" value="http://zhalnin.tmweb.ru"/>

<input type="hidden" name="WMI\_FAIL\_URL" value="http://zhalnin.tmweb.ru"/>

<p align="right" ><input type="submit" name="submitdata" id="confirm-button-next" value="Оплатить" /></p>

</form>';

}

break;

default:

echo '<div id="block-step">

<div id="name-step">

<ul>

<li><a class="active" href="" >1. Корзина товаров</a></li>

<li><span>&rarr;</span></li>

<li><a href="" >2. Контактная информация</a></li>

<li><span>&rarr;</span></li>

<li><a href="" >3. Завершение</a></li>

</ul>

</div>

<p>Шаг 1 из 3</p>

<a href="cart.php?action=clear">Очистить</a>

</div>';

// заголовок таблицы блока

echo '<div id="header-list-cart">

<div id="head1">Изображение</div>

<div id="head2">Наименование товара</div>

<div id="head3">Кол-во</div>

<div id="head4">Цена</div>

</div>';

$sth = DB::getStatement("SELECT \* FROM cart, table\_products WHERE cart.cart\_ip = '{$ip}' AND table\_products.products\_id = cart.cart\_id\_product");

$sth->execute();

$rows = $sth->fetchAll();

if (! empty($rows)) {

$allPrice = 0;

foreach ($rows as $row) {

$int = $row['cart\_price'] \* $row['cart\_count'];

$allPrice = $allPrice + $int;

if (strlen($row['image']) > 0 && file\_exists('uploads\_images/'.$row['image'])) {

$imgPath = 'uploads\_images/'.$row['image'];

$maxWidth = 100;

$maxHeight = 100;

list($width, $height) = getimagesize($imgPath);

$ratioh = $maxHeight / $height;

$ratiow = $maxWidth / $width;

$ratio = min($ratioh, $ratiow);

$width = intval($ratio \* $width);

$height = intval($ratio \* $height);

} else {

$imgPath = 'images/noimages.jpeg';

$width = 120;

$height = 105;

}

// блок с отображение товара в таблице

echo '<div class="block-list-cart">

<div class="img-cart">

<p align="center"><img src="'.$imgPath.'" width="'.$width.'" height="'.$height.'" /></p>

</div>

<div class="title-cart">

<p><a href="" >'.$row["title"].'</a></p>

<p class="cart-mini-features">'.$row["mini\_features"].'</p>

</div>

<div class="count-cart">

<ul class="input-count-style">

<li><p align="center" class="count-minus" iid="'.$row['cart\_id'].'">-</p></li>

<li><p align="center"><input id="input-id'.$row['cart\_id'].'" iid="'.$row['cart\_id'].'" class="count-input" maxlength="3" type="text" value="'.$row["cart\_count"].'" /></p></li>

<li><p align="center" class="count-plus" iid="'.$row['cart\_id'].'">+</p></li>

</ul>

</div>

<div id="tovar'.$row['cart\_id'].'" class="price-product"><h5><span class="span-count">'.$row["cart\_count"].'</span> x <span>'.groupPrice($row["cart\_price"]).' руб</span></h5><p price="'.$row['cart\_price'].'">'.groupPrice($int).' руб</p></div>

<div class="delete-cart"><a href="cart.php?id='.$row["cart\_id"].'&action=delete" ><img src="images/bsk\_item\_del.png" /></a></div>

<div id="bottom-cart-line"></div>

</div>';

}

echo '<h2 class="itog-price" align="right">Итого: <strong>'.groupPrice($allPrice).'</strong> руб</h2>

<p align="right" class="button-next"><a href="cart.php?action=confirm">Далее</a></p>';

} else {

echo '<h3 id="clear-cart" align="center">Корзина пуста</h3>';

}

break;

}

?>

</div>

<?php include('include/block-footer.php'); ?>

</div>

</body>

</html>

<?php

} catch(PDOException $ex) {

throw new Exceptions($ex);

}

?>