

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление 09.03.02 «Информационные системы и технологии»
подготовки

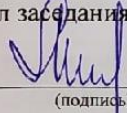
БАКАЛАВРСКАЯ РАБОТА

на тему

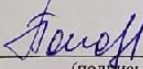
**Web-приложение магазина с
анализом конверсий страниц сайта**

Студент	 14.06.21 (подпись, дата)	Суворов Максим Дмитриевич (ФИО полностью)
Руководитель	 14.06.21 (подпись, дата)	Попова Н.А. (фамилия, инициалы)
Нормоконтролёр	 21.06.21 (подпись, дата)	Такташкин Д.В. (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой  (подпись) Макарычев П.П.
(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК  (подпись) Попова Н.А.
(фамилия, инициалы)

Пенза, 2021

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
П.П. Макарычев
« » 2021 г

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА

1. Студент Суворов Максим Дмитриевич, гр. 17ВИ1 факультета ФВТ
2. Тема работы Web-приложение магазина с анализом конверсий страниц сайта
- Тема утверждена приказом ПГУ № _____ от " ____ " _____ 20 ____
3. Руководитель работы к.т.н доцент., Попова Н.А
4. Задание на работу (назначение разработки, исходные данные и т.п.)

Назначение: предоставление возможности покупки товаров и услуг

Функции системы :

1. Предоставление пользователю возможности выбрать и заказать товар
2. Анализ конверсий страниц сайта
3. Анализ пользовательский действий

Технология разработки: ООП, клиент-сервер

Языки программирования: Java 8, JavaScript, Python

Среды разработки : JetBrains IntelliJ IDEA, JetBrains WebStorm

5. Перечень подлежащих разработке вопросов

1. Анализ предметной области
2. Анализ требований на разработку
3. Проектирование системы
4. Реализация системы

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области	5%	14.04.21 20.04.21	<i>Григорьев</i>
2	Анализ требований на разработку	5%	20.04.21 27.04.21	<i>Григорьев</i>
3	Проектирование программных средств	15%	27.04.21 06.05.21	<i>Григорьев</i>
4	Разработка программных средств	25%	06.05.21 14.05.21	<i>Григорьев</i>
5	Тестирование программных средств	20%	14.05.21 26.05.21	<i>Григорьев</i>
6	Оформление пояснительной записки	30%	26.05.21 13.06.21	<i>Григорьев</i>

Дата выдачи задания: 28.12.2020

Руководитель бакалаврской работы *Григорьев* *Григорьев И.А.*
(подпись, дата)

Задание к исполнению принял студент *Сид* 28.12.2020
(подпись, дата)

Работу к защите допустить
Декан факультета *Л.Р. Григорьев*
(подпись, дата) 24.06.2021

Реферат

Пояснительная записка содержит 90 листов, 33 рисунка, 32 использованных источников, 13 таблиц и 3 приложения.

WEB-МАГАЗИН, АНАЛИЗ КОНВЕРСИЙ, JAVA, JAVASCRIPT, INTELLIJ IDEA, PYNTON, SPRING, КЛИЕНТ-СЕРВЕР.

Объектом исследования являются программные обеспечения для клиента магазина и бизнес-аналитика.

Предмет исследования – технология разработки web-приложения магазина с возможностью проведения анализа конверсий страниц сайта.

Цель работы: реализовать web-приложение для магазина с возможностью проведения анализа конверсий страниц сайта.

Технологии разработки: ООП, Java, MVC, Python, Spring.

Результат работы: спроектировано и разработано web-приложение для магазина с возможностью проведения анализа конверсий страниц сайта.

					ПГУ 09.03.02–08БР171.19 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	«Web-приложение магазина с анализом конверсий страниц сайта»	Лит.	Лист	Листов
Разраб.		Суворов М.Д.		14.06.21				
Провер.		Попова Н.А.		14.08.21			4	90
Н. Контр.		Гакташкин Д.В.		21.08.21				
Утверд.		Макарычев П.П.			Пояснительная записка	Группа 17ВИ1		

Содержание

Введение.....	7
1 Анализ предметной области и постановка задачи на разработку	9
1.1 Состояние исследований разработки web-приложений.....	9
1.2 Актуальность разработки	9
1.3 Цели и задачи сравнительного анализа аналогов	10
1.4 Обзор аналогов.....	12
1.5 Результаты сравнительного анализа.....	19
1.6 Критериальная оценка аналогов	20
1.7 Постановка цели и задачи на разработку	21
1.7.1 Постановка цели на разработку	21
1.7.2 Аппаратное и программное окружение разрабатываемой системы ...	22
1.7.3 Постановка задачи на разработку системы.....	23
1.8 Планирование разработки и расчет бюджета	24
2 Разработка требований к системе	27
2.1 Функциональные требования	27
2.2 Требования к программной архитектуре	30
2.2.1 Требования к общей архитектуре системы.....	30
2.2.2 Требования к архитектуре серверного приложения	32
2.3 Проектирование логической схемы базы данных.....	34
2.4 Обоснование выбора СУБД.....	35
2.5 Обоснование выбора средств разработки	36
2.5.1 Обоснование выбора языковых средств разработки	36
2.5.2 Обоснование выбора фреймворков и библиотек	39
2.5.3 Обоснование выбора инструментальных средств.....	42
3 Разработка web-магазина с анализом конверсий страниц сайта	43
3.1 Разработка физической схемы базы данных	43
3.2 Разработка структуры серверного приложения	44
3.3 Спецификация методов серверного приложения.....	47
4 Руководство пользователя	49

	6
4.1	Описание интерфейса web-магазина 49
4.2	Описание интерфейса бизнес-аналитика 52
5	Тестирование приложения..... 58
5.1	Выбор методологии тестирования ПО 58
5.2	Тестирование приложения..... 59
5.3	Модульное тестирование 61
5.4	Метрики кода 62
	Заключение 64
	Список использованных источников 65
	Приложение А. Глоссарий 68
	Приложение Б. Листинг программы 71
	Приложение В. Тестирование программы 87

Введение

В современном мире все большую популярность приобретает электронная коммерция. Эта сфера экономики включает в себя все финансовые и торговые транзакции, осуществляемые при помощи компьютерных сетей, и бизнес-процессы, связанные с проведением таких транзакций.

Для реализации бизнес процессов, связанных с электронной коммерцией в частном случае используются информационные системы, представляющие собой сайт-магазина в сети Интернет. Такой web-ресурс в общем случае позволяет решать основные задачи связанные:

- оплатой;
- регистрацией;
- анализом предпочтений потребителя;
- заказом товаров или услуг.

Актуальность выбранной темы связана с тем, что для осуществления краткосрочного и долгосрочного планирования бизнесу требуется постоянно собирать статистику о текущем положении на рынке. Актуальные данные позволят быстро принимать управленческие решения, для корректировки бизнес-плана. Одним из важнейших инструментов продвижения продуктов и бизнеса является интернет-реклама. Для улучшения качества и эффективности рекламы производится анализ маркетинговой деятельности путем проведения А/В-тестирования [1].

В рамках выполнения выпускной квалификационной работы объектом исследования являются программные обеспечения для клиента магазина и бизнес-аналитика.

Предметом исследования является технология разработки web-приложения магазина с возможностью проведения анализа конверсий страниц сайта.

Главной целью разработки является создание системы web-магазина с возможностью анализа пользовательских действий и конверсий страниц сайта. Для достижения поставленной цели необходимо решить следующие задачи:

- выполнить анализ предметной области;
- провести сравнительный анализ существующих систем-аналогов;
- на основе анализа выполнить постановку задачи на разработку web-магазина с возможностью анализа пользовательских действий;
- определить функционал приложения с использованием языка UML;
- разработать сайт web-магазина с возможностью анализа пользовательских действий в соответствии с поставленной задачей;
- провести тестирование ПО.

Существующие системы-аналоги требуется тщательно изучить и сделать выводы о достоинствах и недостатках.

1 Анализ предметной области и постановка задачи на разработку

1.1 Состояние исследований разработки web-приложений

Для анализа актуальности проблемы разработки web-приложений следует рассмотреть состояние исследований в данной области.

В статье Максим Гапчука «Как правильно собирать и использовать данные о действиях пользователей на сайте» рассматриваются способы сбора пользовательских данных и организации их анализа [2]. Автор рассматривает использование инструмента Google Analytics при принятии правильных бизнес-решений. Важную роль в анализе действий пользователя играет анализ макро- и микроконверсий, т.е анализ необходимых шагов в воронке продаж, а также любых действий, которое предпринимает пользователь, прежде чем начинает интересоваться продуктом или брендом. Эти данные дают возможность увеличить CPA (Cost Per Action, стоимость полезного действия с различных рекламных источников, кампаний и ключевых слов) рекламы и качество ее содержания.

Компания «beseller» специализирующаяся на создании web-сервисов интернет магазинов рассматривает web-аналитику как процесс необходимый каждой компании [3]. Внимание акцентируется на получение оперативных данных о поведении пользователя на сайте магазина, что позволяет быстро менять функциональные требования к товарам, услугам, рекламе и web-ресурсу в целом. Для осуществления сбора статистики о всех пользовательских действиях, требуется многофункциональный инструмент, позволяющий гибко настраивать параметры сбора статистики.

1.2 Актуальность разработки

Актуальность разработки web-приложений с возможностью анализа пользовательских действий, обусловлено тем, что аналогичных инструментов существует не так много. Существующие системы тяжело настраиваемые и в

большинстве из них отсутствует возможность определения местоположения и идентификации пользователя.

Актуальность создания web-решения для интернета магазина, определяется также востребованностью данных ресурсов на рынке. Как показано в исследовании «Ассоциация компаний интернет-торговли» сектор интернет-торговли в январе – феврале 2021 года увеличился по сравнению с аналогичным периодом годом ранее на 44% и достиг 431,3 млрд рублей [4].

Ассоциация отмечает устойчивый рост внутреннего рынка онлайн-торговли на фоне роста конкурентной активности и качества услуг российских компаний, а также локализации бизнеса зарубежных игроков, что повлекло активное развитие внутреннего рынка труда и рост налоговых поступлений.

Web-приложение магазина с возможностью анализа пользовательских действий актуально для малого бизнеса осуществляющего основные продажи через интернет и использующие средства интернет рекламы для продвижения товаров и услуг.

Внедрение разработанной системы позволит анализировать потребности потребителя, оценивать успешность рекламной политики компании, измерять KPI (Key Performance Indicators, ключевые показатели эффективности), а также CPS (Cost-per-Sale, оплата за продажу) [5]. В дальнейшем в системе будет возможность интеграции с существующими сервисами агрегации аналитики.

1.3 Цели и задачи сравнительного анализа аналогов

Сравнительный анализ современных web-приложений позволит определить наиболее используемые систему, а также получить представление о том, насколько они распространены и актуальны.

Можно выделить следующие цели проведения анализа систем:

- выявление достоинств;
- выявление недостатков;
- выявление характерных базовых функций;

- выявление характерных функциональных особенностей.

По итогам анализа главной задачей будет определение обязательных функций, а также уникальных возможностей, которые расширят функционал реализуемой системы и приведут к ее высокой конкурентоспособности.

На сегодняшний день существует большое количество средств, позволяющих создавать интернет-магазины с возможностью анализа данных.

Ниже будут приведены некоторые из них:

- Битрикс24;
- Wordpress;
- Firebase;
- WIX;
- Tilda Publishing;
- Webflow;
- Jimdo;
- Webnode;
- Vigbo;
- uKit.

Часть этих средств ориентировано только на создания web-формы с минимальным возможным анализом. Для сравнительного анализа выбраны системы предоставляющие возможность комплексного анализа действий пользователя. Выбранные сервисы предоставляют бизнес-аналитику и инфографику по выбранным критериям анализа. К этим системам относятся:

- Битрикс24;
- Wordpress;
- Firebase;
- WIX;
- Tilda Publishing.

Представленные средства используют разный способ монетизации и тарифные планы использования.

Для дальнейшего анализа были выбраны наиболее популярные инструменты создания web-приложений с возможностью анализа данных. Представленные системы являются наиболее распространенными при решении задач связанных с созданием web-ресурсов и анализом пользовательских данных.

1.4 Обзор аналогов

Битрикс24 – сервис для управления бизнесом и анализом основных бизнес параметров. ПО предоставляет широкие возможности по созданию web-приложений в том числе интернет-магазинов. Данное ПО позволяет создать собственный сайт интернет магазина и внедрить шаблонную аналитику поведения пользователей. Конкурентным преимуществом сервиса является встроенная система управления взаимоотношения с клиентами. Эта функциональная особенность позволяет автоматизировать процесс продаж и оформления интернет-заказов. Конструктор сайтов Битрикс24 предлагает набор шаблонов сайтов-магазинов, позволяющий быстро реализовать бизнес-идею. За дополнительную плату компания предоставляет услуги создания и внедрения специализированных сайтов-магазинов и других web-ресурсов. На рисунке 1 представлен интерфейс системы.

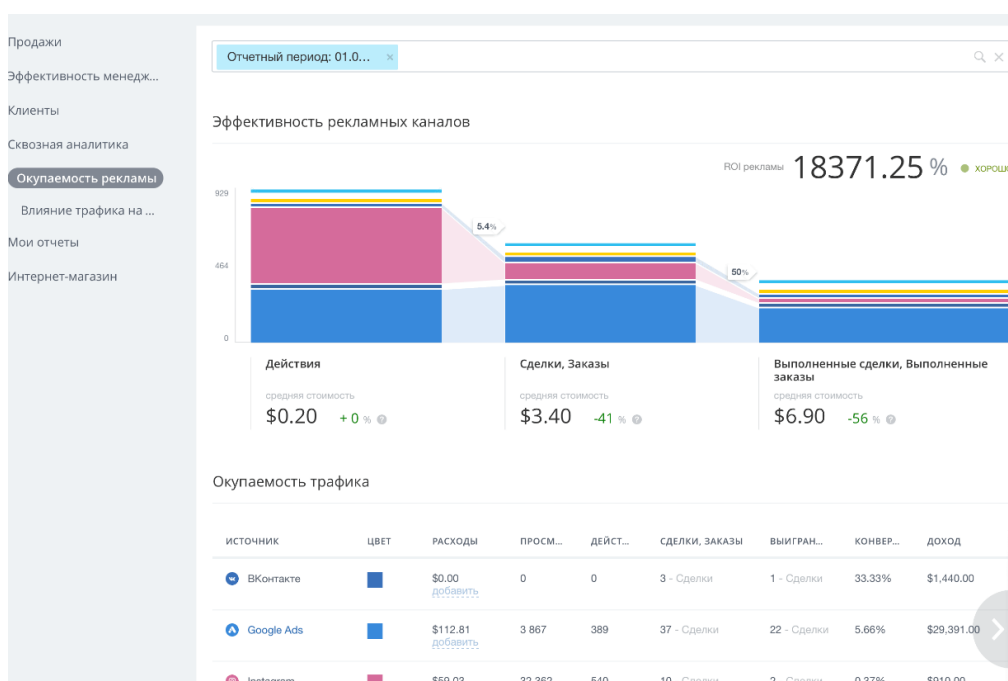


Рисунок 1 – Интерфейс Битрикс24

Wordpress – бесплатное свободно распространяемое программное обеспечение, ориентированное на создание сайтов. Это средство позволяет почти без настройки реализовывать задачи по созданию web-ресурсов. Система интуитивна понятна и проста. Система Wordpress использует PHP и MySQL и лицензируется под GPLv2, что позволяет использовать систему под управлением любой ОС. Отличительной чертой данной системы является высокая производительность созданных на ней решений. По данной система разработана полная техническая документация, что позволяет увеличить скорость разработки web-ресурсов и снизить риски совершения программных ошибок. Wordpress предоставляет более 58 371 расширения для реализации почти любых поставленных задач [6, 7]. Для осуществления отслеживания пользовательских действий используется расширение YOAST SEO, позволяющее собирать статистику по использованию сайта. На рисунке 2 представлен интерфейс YOAST SEO.

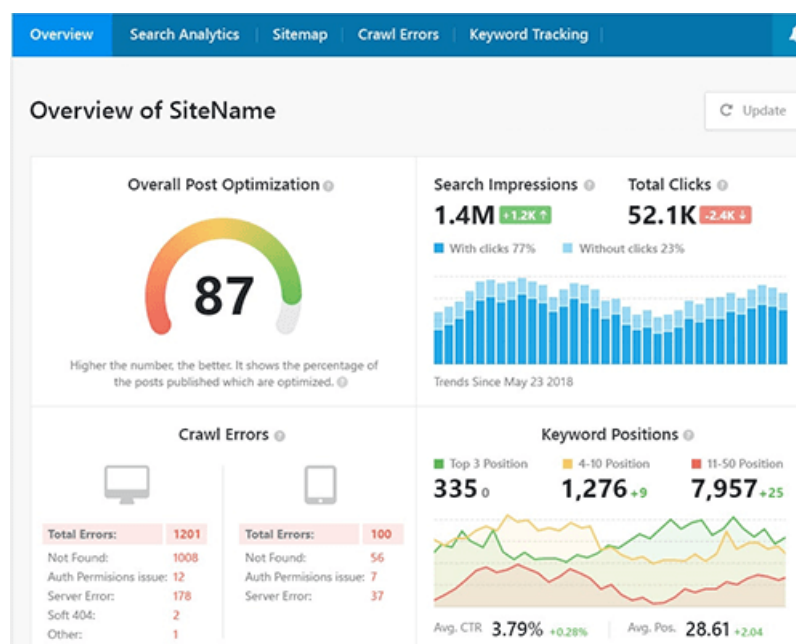


Рисунок 2 – Интерфейс YOAST SEO

Firebase – облачное средство, позволяющее разработчикам приложений хранить и синхронизировать данные между несколькими клиентами. Ключевой особенностью системы является возможность интегрироваться не только с web-приложениями, но и с Android и Ios клиентами. ПО предоставляет широкий спектр возможностей для анализа активности на

сайте, с возможностью добавления пользовательских метрик [8]. Сервис предоставляет возможность хостинга сайта и серверной части на своих серверах. Дополнительной функциональностью, которую предлагает сервис, является возможность использования серверов Firebase для машинного обучения и проведения аналитики больших данных. Для реализации аутентификации на web-ресурсе, Firebase дает возможность за дополнительную плату использовать собственное API аутентификации. На рисунке 3 представлен интерфейс отображающий статистику по конверсиям и посещениям приложения.

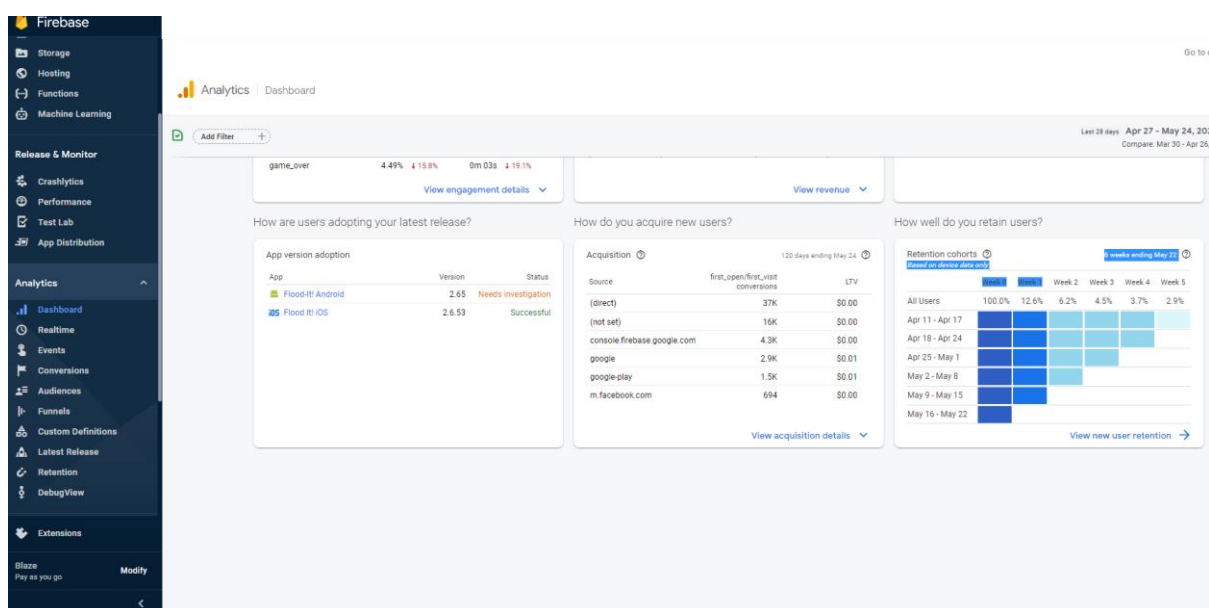


Рисунок 3 – Статистика Firebase по конверсиям приложения

Tilda Publishing – средство создания web-ресурсов, не требующий глубоких знаний в программировании. Система позволяет создавать сайты, интернет-магазины, посадочные страницы, блоги и email-рассылки. Tilda работает по модели Freemium: на сервисе есть возможность использовать бесплатный план для создания сайтов и несколько вариантов подписки, которые отличаются функциональностью: например, количеством доступных для создания сайтов, возможностью экспорта, доступом к API [9]. Средство предоставляет возможность принимать онлайн платежи. Для этого в системе предусмотрена интеграция с такими системами как:

- Яндекс. Деньги;

- PayPal;
- Visa;
- MasterCard;
- МИР.

Для проведения анализа по посещению сайта требуется подключение стороннего сервиса Google Analytics, результат использования расширения представлен на рисунке 4.

Явным конкурентом Tilda Publishing является схожий сервис по созданию сайтов и интернет-магазинов WIX. WIX – web-сервис, позволяющий создавать и размещать сайты. Данное ПО имеет схожую модель монетизации с Tilda, но гораздо более высокий тарифный план для платных услуг. Wix предоставляет сотни настраиваемых шаблонов веб-сайта и HTML5-редактор, работающий по принципу drag-and-drop, который включает в себя приложения, графику, галереи изображений, видео, шрифты и многое другое. Шаблоны настраиваются: добавляются новые функции и медиа, меняются стиль, цвета, тексты, фоновые изображения, кнопки и др.

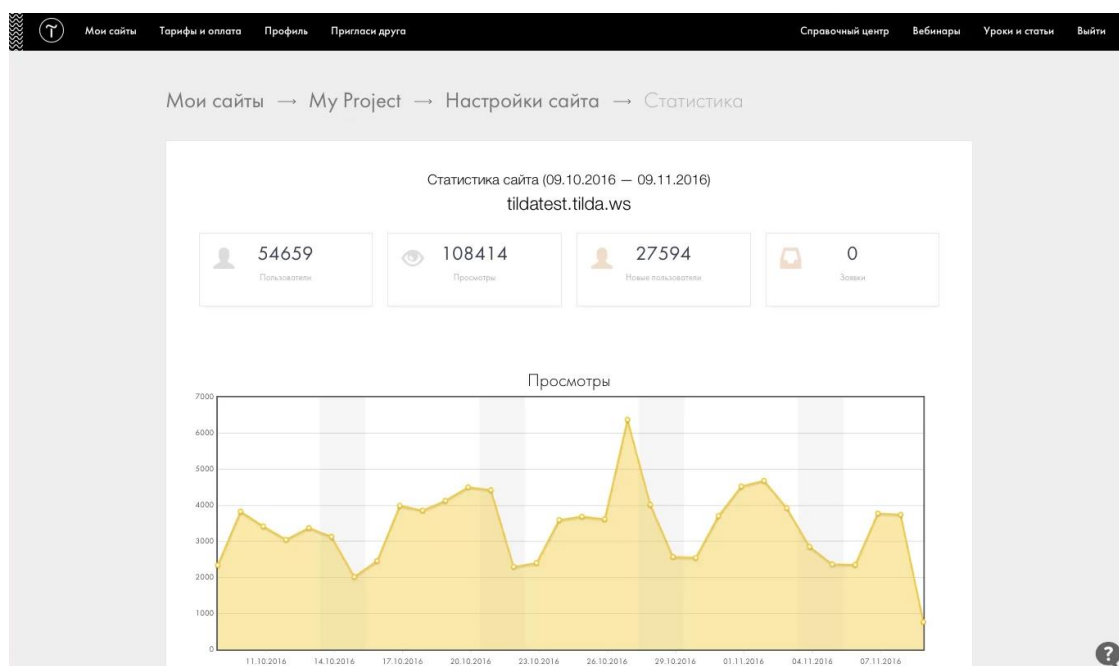


Рисунок 4 – Статистика Tilda по посещениям сайта

Ключевыми особенностями WIX являются:

- бесплатный хостинг;

- создание всплывающих окон;
- размещение данных в соцсетях;
- поисковая оптимизация;
- защита страниц;
- модуль подключения Яндекс.Кассы;
- оптимизация для мобильных устройств.

Стандартный набор инструментов WIX не поддерживает возможности анализа трафика. Для организации отслеживания пользовательских действий используется расширение Visitor Analytics, позволяющее предоставить менеджеру или бизнес-аналитику отчет о интернет-трафике на сайте. Для отслеживания переходов из социальных сетей используется платное расширение Pixel Conversion Tracking. Интерфейс для аналитика, предоставляемый системой Pixel Conversion Tracking, представлен на рисунке 5.

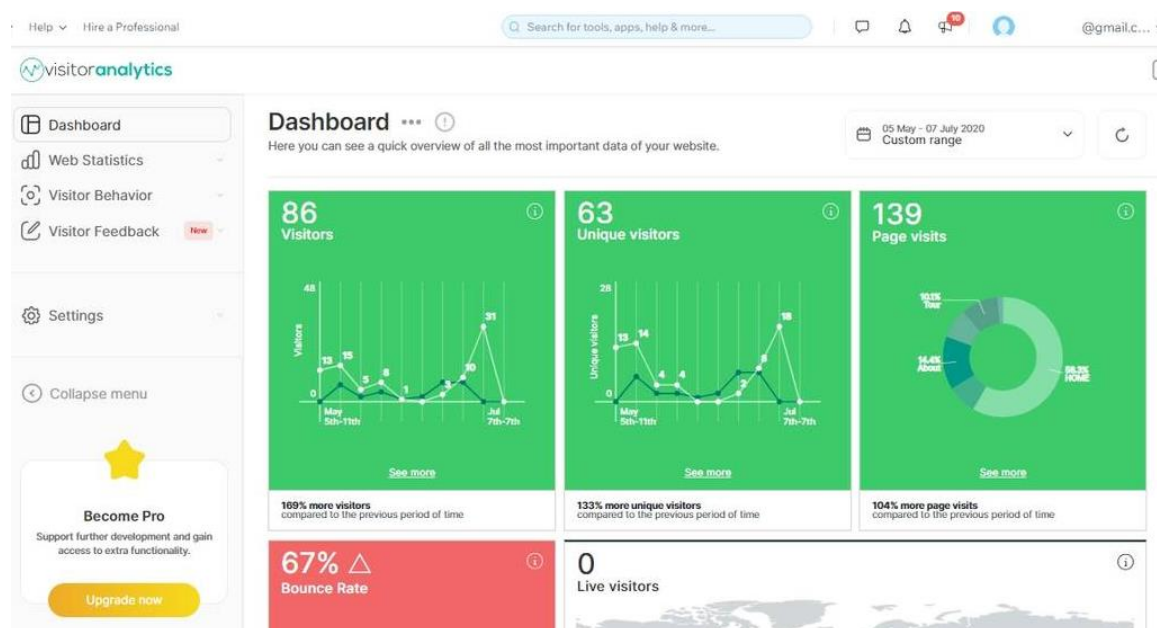


Рисунок 5 – Статистика, предоставляемая расширением Visitor Analytics

Для оценки, анализа и сравнения инструментальных средств были выделены следующие критерии оценки:

- графический интерфейс пользователя – интерфейс, предоставляемый пользователю-клиенту web-ресурса, позволяющий осуществить операции по заказу товара или услуги;

- API сервера – готовая серверная часть приложения для взаимодействия с клиентом и обработки HTTP-запросов и отправки;
- отслеживание трафика – возможность агрегировать информацию о ресурсах с которых переходят клиенты;
- конструктор сайта – возможность создать web-страницу для публикации;
- анализ конверсий – возможность анализа числа посетителей сайта, выполнивших на нем какие-либо целевые действия, к общему числу посетителей сайта;
- A/B тестирование – возможность проведения исследований по выявлению более востребованных товаров и услуг;
- поведенческие паттерны пользователя – возможность отслеживать поведение человека на сайте;
- отслеживание гео-данных – сбор и агрегация данных о местоположении клиента;
- регистрация и аутентификация – возможность зарегистрировать пользователя по e-mail и паролю и предоставить доступ к личному кабинету по этим же данным.

Результаты анализа функциональных возможностей выделенных средств создания web-приложений и анализа пользовательских действий представлены в таблице 1.

Таблица 1 – Результирующая таблица анализа систем по выбранным критериям

Функции	Системы				
	Битрикс24	Wordpress	Firebase	Tilda Publishing	WIX
Графический интерфейс пользователя	+	+	-	+	+
API сервера	+	+	+	+	+
Отслеживание трафика	+	+	+	+	+
Конструктор сайта	+	-	-	+	+
Анализ конверсий	+	-	+	-	-
A/B тестирование	+	-	+	-	-
Поведенческие паттерны пользователя	+	-	+	-	-
Отслеживание гео-данных	-	-	+	-	-
Регистрация и аутентификация	+	+	+	+	+
Доступность	Платная, есть бесплатная версия с ограничениями на анализ данных и количеством трафика	Бесплатная, есть платные расширения	Бесплатная, для увеличения функционала требуется платная подписка	Бесплатная, для увеличения количества страниц сайта требуется подписка	Платная, есть бесплатная версия с рекламой WIX

На основе результирующей таблицы наглядно определены сильные и слабые стороны систем создания web-ресурсов с возможностью анализа пользовательских действий.

1.5 Результаты сравнительного анализа

В результате анализа было выявлено, что графический интерфейс для пользователя предоставляют все описанные системы, кроме Firebase. Все описанные системы предоставляют серверную часть для обработки пользовательских запросов. Серверная часть в системе Wordpress предоставляется как исходный код на языке PHP, что усложняет развертывания и внедрение. Остальные системы реализуют серверную часть как набор общедоступных методов взаимодействия с сервером.

Отслеживание трафика доступно во всех представленных системах, но с разным уровнем детализированности. Firebase предоставляет больше всего данных о посещении пользователя, дает возможность отследить влияние рекламы на посещения и организовать A/B-тестирование. Минимальный функционал предоставляет Tilda, позволяя просмотреть только количество посещений за промежуток времени.

Конструктор форм сайта предоставляется платформами: Битрикс24, WIX, Tilda. Системы предоставляют возможность создать простую форму сайта разной степени адаптивности. Наиболее плохо адаптированные под разные типы устройств страницы предоставляет Битрикс24. Высокая степень адаптивности созданного сайта показывает система WIX.

Функциональную возможность отслеживания поведенческих паттернов пользователей предоставляет только системы Firebase и Битрикс24, при этом система Битрикс24 эта возможность доступна только по платной подписке.

Для определения целевой аудитории требуется данные о географическом расположении клиента. Такую функциональную возможность предоставляет только система Firebase. Остальные системы не предлагают никакой возможности для простейшего сбора информации о географическом расположении посетителя сайта.

В большинстве рассмотренных систем, присутствуют стандартные функции, которые необходимо поддерживать и в реализуемой системе. К ним относятся:

- API сервера;
- графический интерфейс пользователя;
- регистрация и аутентификация.

Кроме обязательных функций, в системах реализован набор дополнительных функциональных возможностей таких как:

- анализ конверсий;
- А/В тестирование;
- поведенческие паттерны пользователя;
- отслеживание гео-данных;
- отслеживание трафика.

Данные функциональные особенности делают описанные программные продукты конкурентоспособными и эффективными.

1.6 Критериальная оценка аналогов

Для проведения критериальной оценки ПО, следует выделить основные метрики оценки и веса за каждый критерий. Для наиболее точной оценки требуется выбрать значимые критерии и задать оценку в порядке возрастания.

Критерий оценки и максимальный возможный балл за критерий представлен в таблице 2.

Таблица 2 – Максимальные и минимальные баллы

Критерий	Минимальная оценка	Максимальная оценка
Графически интерфейс	0	0,5
API сервера	0	0,8
Отслеживание трафика	0	0,7
Конструктор сайтов	0	0,2
Анализ конверсий	0	0,9
А/в тестирование	0	1
Отслеживание гео-данных	0	1

Для выявления, наиболее подходящего ПО для создания и web-ресурсов и анализа пользовательских действий сопоставим веса с каждым продуктом.

Результаты оценки программных средств представлены в таблице 3.

Таблица 3 – Оценка ПО по выбранным критериям

Критерий	Битрикс24	Firebase	Tilda	Wix	Wordpress
Графически интерфейс	0,41	0	0.45	0.49	0.3
API сервера	0,5	0,5	0,5	0,5	0,3
Отслеживание трафика	0,47	0,7	0,5	0,4	0,3
Конструктор сайтов	0,2	0	0,2	0,2	0
Анализ конверсий	0,8	0,9	0	0	0
A/b тестирование	0,5	1	0	0	0
Отслеживание гео-данных	0	0,5	0	0	0

Для определения, наиболее подходящего ПО все баллы за выбранные метрики были просуммированы. Результаты проведения критериального анализа представлены в таблице 4.

Таблица 4 – Результат критериального анализа

Система	Суммарный балл
Битрикс24	2,28
Firebase	3,6
Tilda	1,2
Wix	1,1
Wordpress	0,6

В ходе анализа было выявлено, что наиболее, функциональной системой по выбранным критериям является Firebase. Firebase позволяет реализовать наибольшее число функций для анализа пользовательского поведения.

Наименьшую оценку по выбранным критерию получила система Wordpress. Система реализует минимальное возможное число функций для анализа поведения и не имеет конструктора для создания страниц сайта.

1.7 Постановка цели и задачи на разработку

1.7.1 Постановка цели на разработку

Целью разработки является создание универсального web-приложения магазина с возможностью анализа конверсий и пользовательских действий на

сайте. Система должна позволять бизнес-аналитикам и менеджерам анализировать действия пользователя на сайте, а также проводить A/B-тестирование для выбранных товаров и услуг. Собранные статистики по пользовательским действиям должны визуализироваться и помогать в принятии правильных бизнес-решений. Web-приложение должна обеспечивать регистрацию и аутентификацию пользователя.

Пользователей системы можно разделить на две категории: бизнес-аналитик и клиент. В рамках системы можно четко разграничить две представленные роли.

Клиент – человек интересующийся товаром или услугой предоставляемой компанией. Для клиентов предоставляется интерфейс выбора интересующего товара и его покупке. При этом пользователь должен зарегистрироваться в системе для покупки товара. Данные задачи могут быть выполнены клиентом из любого поддерживаемого браузера, а также его мобильной версии.

Бизнес-аналитику – управляющей менеджер отвечающей за продвижение товаров и услуг, предоставляемых компанией. Для пользователя предоставляется статистика активности на сайте, а также данные о конверсиях и поведении пользователя на странице web-ресурса. Для данного типа пользователя предоставляется отдельное приложение, позволяющее выполнять требуемые задачи.

1.7.2 Аппаратное и программное окружение разрабатываемой системы

Для работы приложения у разных видов пользователя предоставляются разные требования к ЭВМ.

Для клиента требуется иметь ПК или мобильный телефон с поддержкой следующих браузеров:

- Все браузеры, основанные на платформе Chromium (Google Chrome, Yandex browser);
- Firefox;

- Safari;
- IE 10.

Для работы серверной части требуется ЭВМ со следующими программно-аппаратным средствам:

- Операционная система: Window 7 и выше, любой дистрибутив Debian 18.2 и выше;
- Наличие СУБД MySQL Server;
- Java Runtime Environment 1.8;
- 512 мб ОЗУ;
- свободное место не менее 5 Гб на жестком диске.

Для работы приложения аналитика требуется подключение к интернету и установленный Python 3.8 и выше.

1.7.3 Постановка задачи на разработку системы

Приложение для клиента должно включать следующие функциональные возможности:

- отображение всех доступных товаров и услуг;
- отображение выбранного товара;
- возможность зарегистрироваться;
- возможность авторизоваться;
- возможность заказать товар.

Приложение для бизнеса-аналитика должно обеспечивать выполнение следующих задач:

- анализ конверсий страниц сайта;
- возможность анализа поведения пользователя на странице сайта;
- возможность проведения А/В-тестирования товаров или услуг;
- возможность анализа успешности проведения рекламных акций.

Оба приложения должны предоставлять дружелюбный интерфейс пользователя.

Интерфейс должен быть создан с стиль графического дизайна Material Design [10]. Данный стиль проектирования графических интерфейсов программного обеспечения был разработан компанией Google для обеспечений наиболее комфортного взаимодействия с ПО. Основное преимущество данного стиля проектирования интерфейсов является то, что он учитывает опыт пользователя в использовании других программных продуктов.

Интерфейс должен быть привлекательным, удобным практичным, разумно и понятно скомпонованным и соответствовать нормам эргономики.

1.8 Планирование разработки и расчет бюджета

Планирование разработки и расчет бюджета является важнейшей частью любого проекта. Для планирования разработки приложения выбрана диаграмма Ганта. Диаграммы Ганта позволяют наглядно изобразить составляющие части проекта и разбить его на задания меньшего размера для удобства управления. Диаграммы Ганта помогают командам планировать работу с учетом сроков и правильно распределять ресурсы, а также упрощают создание сложных планов [11].

Для составления дигаммы Ганта был выбран инструмент Microsoft Project. Для разработки программы необходимо нанять команду, состоящую из шести человек.

На рисунке 6 представлен результат заполнения листа ресурсов, используемых в проекте.

	Resource Name	Тип	Единицы измерения материалов	Краткое название	Organiz	Группа	Max. Units	Std. Rate
	DevOps-инджин	Трудовой		D	Customer	Deployment Team	100%	250,00 Р/ч
	SEO	Трудовой		S	Exec	Management	100%	300,00 Р/ч
	Аналитик	Трудовой		A	IT	Аналитик	100%	200,00 Р/ч
	Разработчик	Трудовой		P	IT	Разработчик	400%	200,00 Р/ч
	Testers	Трудовой		T	IT	Testers	100%	250,00 Р/ч
	Менеджер	Трудовой		M	PMO	Project Manager	100%	300,00 Р/ч

Рисунок 6 – Ресурсы проекта

Для планирования необходимо определить календарный план проекта и длительность выполнения отдельных задач. На рисунке 7 представлен результат разбиения проекта на подзадачи.

предоставленные задачи			предоставленные ресурсы			деловые		
Task Name - 34 - S C A L I N G	Count	Factors	Previous Unit Effort	New Unit Effort	Effort	рудозатраты	Длительность	Названия ресурсов
Утверждение сроков на поставку ПО	1	Fixed	8	8	8	8ч	4ч	SEO;Менеджер
Analysis complete	0		0	0	0	0ч	0д	
Design	0		0	0	76	102ч	10,75д	
Ознакомление с предварительных спецификаций	1	Fixed	16	16	16	16ч	2д	Аналитик
Разработка функциональных спецификация к ПО	6	Features	5	5	30	30ч	3,75д	Аналитик
Разработка UI-прототипа на основании спецификаций	2	Prototype	12	12	24	24ч	3д	Аналитик
Утверждение плана на разработку и спецификаций к ПО	1	Fixed	32	6	6	32ч	16ч	SEO;Менеджер
Design complete	0		0	0	0	0ч	0д	
Development	0		0	0	204	370ч	16,67д	
Просмотр тех. Спецификаций	1	Fixed	8	8	8	8ч	0,5д	Разработчик[200%]
Разработка архитектуры приложения	1	Fixed	32	8	8	32ч	2д	Разработчик[200%]
Назначать сотрудников по разработке	1	Fixed	8	8	8	8ч	0,5д	Менеджер[200%]
Разработка	6	Features	38,67	15	90	232ч	9,67д	Разработчик[300%]
Первичное тестирование разработчиками	6	Features	15	15	90	90ч	11,25д	Разработчик
Development complete	0		0	0	0	0ч	0д	
Testing	0		0	0	180	164ч	33,17д	

Рисунок 7 – Перечень задач и ресурсов проекта

После описания задач и ресурсов Microsoft Project позволяет создать диаграмму Ганта (рисунок 8).

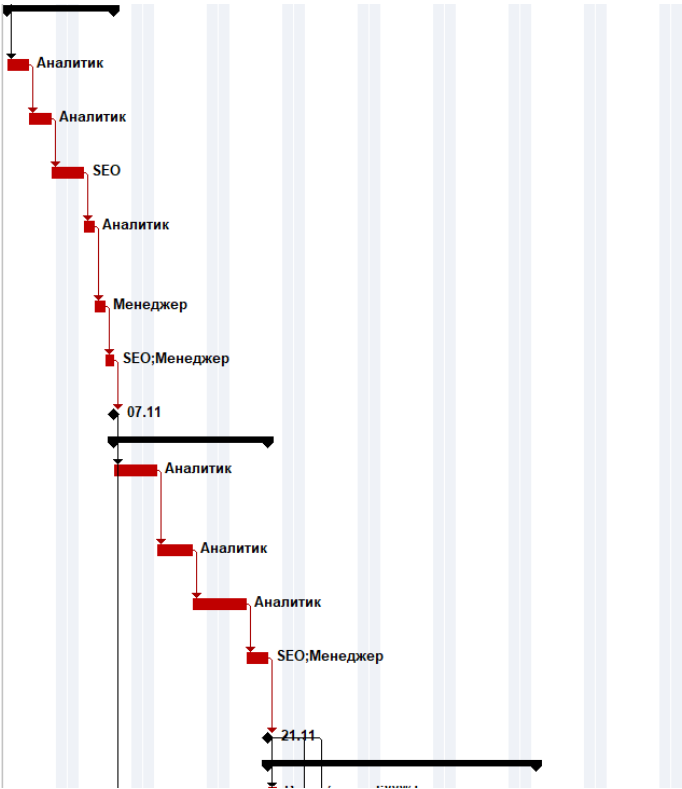


Рисунок 8 – Диаграмма Ганта

Microsoft Project включает набор готовых отчетов и панелей мониторинга, который помогает контролировать общее состояние проекта и проверять соответствие его компонентов графику [12].

Система позволяет на основе диаграммы Ганта и списка ресурсов составлять графики, отражающие информацию о запланированных, оставшихся, фактических, совокупных, базовых затратах и процент завершения проекта. Результат создания отчета по диаграмме Ганта представлен на рисунке 9.

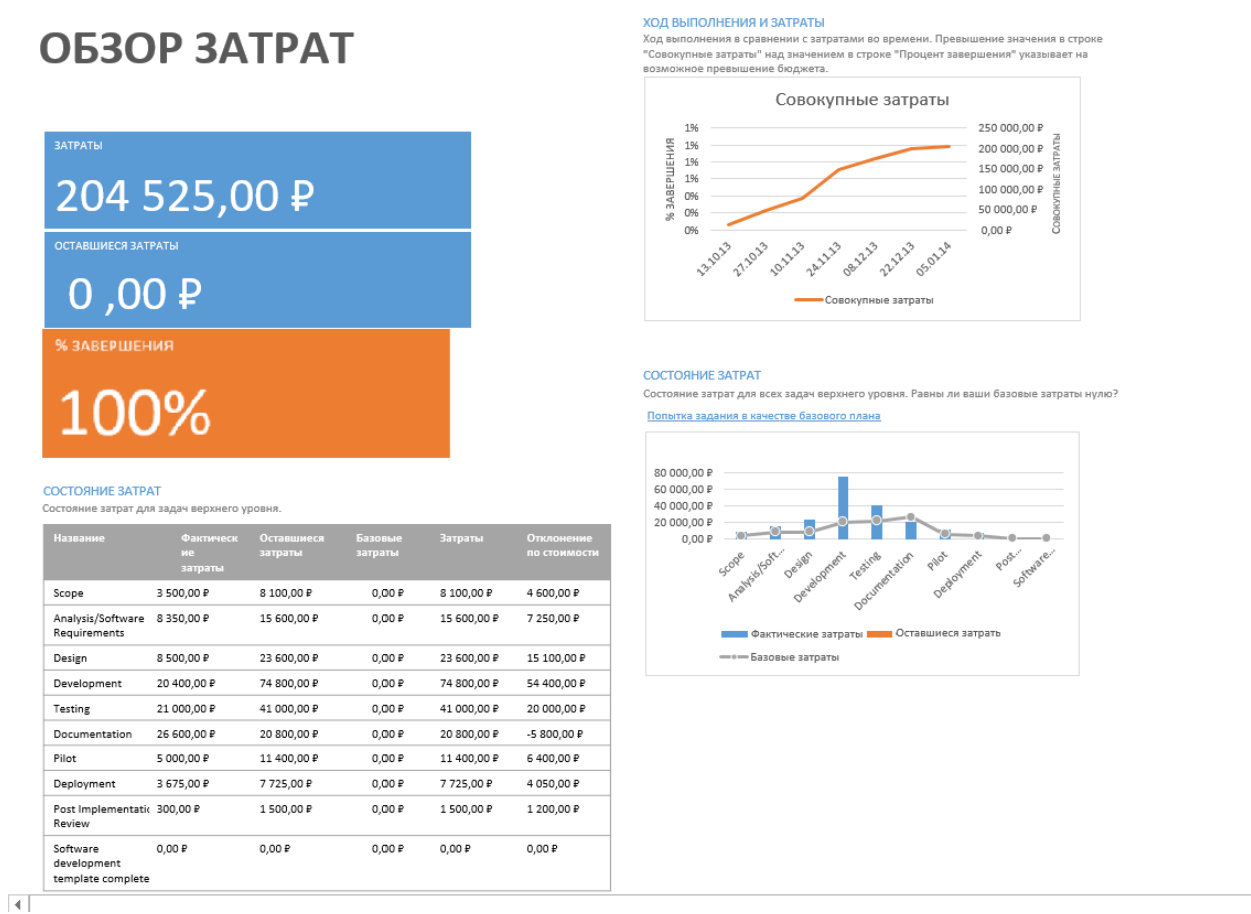


Рисунок 9 – Отчет «Обзор затрат»

Фактические затраты реализации приложения web-магазина с возможностью анализа конверсий страниц сайта составили 204 525,00 рублей, а длительность проекта, с учетом праздников и выходных дней составила 123 дня.

2 Разработка требований к системе

2.1 Функциональные требования

Web-приложение магазина с возможностью анализа пользовательских действий и конверсий должно обеспечивать комфортную работу всех категорий пользователей системы.

Для наглядного представления функций системы использованы диаграммы общецелевого языка визуального моделирования UML. Функциональные возможности приложения представлены UML-диаграммой вариантов использования.

UML – язык графического описания для объектного моделирования в области разработки программного обеспечения и моделирования бизнес-процессов.

Диаграмма вариантов использования – исходный концепт представления ИС в процессе ее проектирования. Основными составляющими данной диаграммы являются: актеры, варианты использования и отношения между ними. При построении диаграммы могут использоваться также общие элементы нотации: примечания и механизмы расширения и включения.

На диаграмме вариантов использования не отображаются особенности внутренней организации системы и специфика программно-аппаратного комплекса средств.

Данная диаграмма предназначена только для определения функциональных требований к системе. При этом учитывается то, что чрезмерная детализация варианта использования не требуется, т.к. вариант использования – это относительно крупный блок функциональности системы.

Диаграмма вариантов использования web-приложению с возможностью анализа пользовательских действий отображена на рисунке 10.

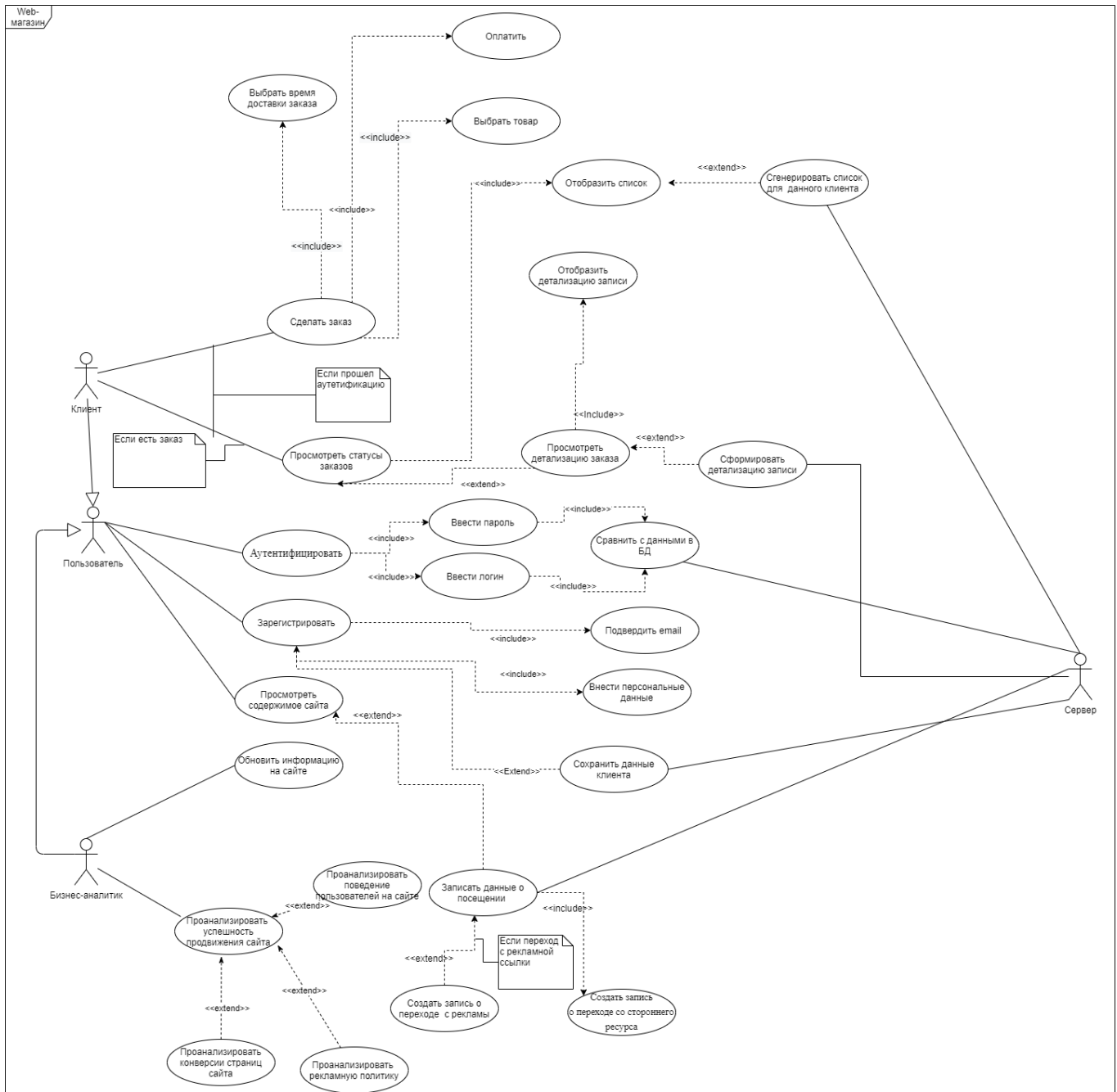


Рисунок 10 – Диаграмма вариантов использования

Краткое описание каждого варианта использования представлено в таблице 5.

Таблица 5 – Краткое описание вариантов использования

Вариант использования	Краткое описание
1	2
Просмотреть содержимое сайта	Позволяет отобразить доступные в данный момент товары и услуги
Зарегистрироваться	Позволяет пользователю создать учетную запись для дальнейшей работы на сайте
Аутентифицироваться	Позволяет пользователю зайти под его учетной записью

Продолжение таблицы 5

1	2
Обновить информацию на сайте	Позволяет бизнес-аналитику добавить новый товар или услугу, а также добавить к ней описание
Сделать заказ	Аутентифицированный пользователь покупает выбранный товар или услугу
Просмотреть статус заказов	Аутентифицированный пользователь просматривает статус заказов, который он купил
Просмотр детализации записи	Аутентифицированный пользователь просматривает историю изменений статуса, стоимость покупки, время и дату последнего изменения
Сгенерировать список заказов	Для аутентифицированного пользователя создается список заказов, которые он приобрел
Сформировать детализацию записи	Для аутентифицированного пользователя по его id формируется вся информация о выбранном заказе
Сравнить с данными в БД	Сервер сравнивает поступившие данные с сохраненными в БД
Сохранить данные клиента	Сервер валидирует данные клиента и сохраняет их
Записать данные о посещении	Сервер сохраняет рекламный идентификатор пользователя и страницу на которую он зашел
Выбрать время доставки заказа	Аутентифицированный пользователь выбирает время доставки купленного товара заказа
Оплатить	Аутентифицированный пользователь оплачивает выбранный товар.
Выбрать товар	Аутентифицированный пользователь выбирает товар из списка для дальнейшей покупки
Отобразить список	Отображение истории заказов аутентифицированного пользователя
Отобразить детализацию записи	Отображение полной информации о заказе
Ввести пароль	Пользователь вводит новый пароль для возможности в дальнейшем зайти в аккаунт
Ввести логин	Пользователь вводит e-mail, который используется как логин
Подтвердить e-mail	Пользователь вводит код-подтверждения из e-mail сообщения
Внести персональные данные	Пользователь вводит ФИО в форме регистрации
Проанализировать успешность продвижения сайта	Бизнес-аналитик используя инструменты ИС проводит анализ продвижения сайта
Проанализировать поведение пользователей на сайте	Бизнес-аналитик используя инструменты ИС строит карту перемещений пользователя на сайте

Окончание таблицы 5

1	2
Проанализировать конверсии страниц сайта	Бизнес-аналитик используя инструменты ИС проводит анализ числа посетителей сайта, выполнивших на нем какие-либо целевые действия, к общему числу посетителей
Проанализировать рекламную политику	Бизнес-аналитик используя инструменты ИС проводит анализ количества покупок, совершенных после проведения рекламной акции
Создать запись о переходе с рекламного объявления	При переходе с рекламной ссылки осуществляется запись рекламного агрегатора
Создать запись о переходе со стороннего ресурса	При переходе со стороннего сервиса, создается запись со ссылкой исходного перехода

Пользователь, может зарегистрировать несколько акантов на разные e-mail почты. Для удовлетворения мер безопасности повторная отправка кода с подтверждением e-mail возможна только через установленное количество времени.

2.2 Требования к программной архитектуре

2.2.1 Требования к общей архитектуре системы

Для реализации ИС была выбрана трехзвенная клиент-серверная архитектура [13]. Она представляет собой дальнейшее развитие технологии «клиент-сервер». В трехзвенной архитектуре вся бизнес-логика, выделяется в отдельное звено, называемое сервером приложений. При этом клиентским приложениям остается лишь пользовательский интерфейс. Так, в качестве клиентского приложения представлено web-сайт, отображаемое в браузере пользователя. Трехзвенная архитектура может быть расширена до многозвенной путем выделения дополнительных серверов, каждый из которых будет представлять собственные сервисы и пользоваться услугами прочих серверов разного уровня. Данная архитектура предоставляет следующие преимущества:

- сервер приложений имеет возможность управления пулом соединений с базой данных, в связи с чем может быть уменьшено число одновременно установленных соединений с сервером баз данных (одно соединение может обслуживать нескольких клиентов);
- снижаются требования к конфигурации и производительности рабочих станций, что снижает затраты на построение системы;
- позволяет повысить безопасность системы за счет сокрытия реализации бизнес-логики в серверной части приложения.

В web-приложении с возможностью анализа конверсий и пользовательских действий роль клиентов играют приложение покупателя и приложение бизнес-аналитика. Работа по выборке, сортировке и агрегации данных полностью выполняется серверной частью приложения.

Детально архитектура системы представлена на рисунке 11. На диаграмме представлена web-приложение с возможностью анализа конверсий, которая состоит из приложения бизнес-аналитика, приложение-пользователя, серверного приложения, обрабатывающего HTTP-запросы и реляционной базы данных. Серверное приложение представляет собой программу работающую на выделенном ЭВМ с предустановленной средой исполнения Java Virtual Machine.

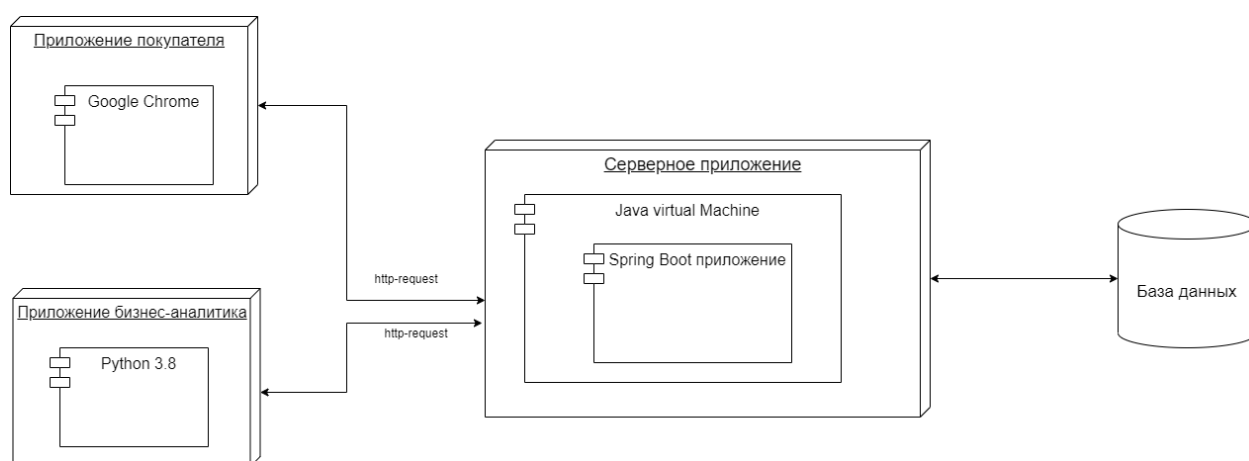


Рисунок 11 – Архитектура системы

Серверное приложение позволяет сохранить одно подключение к базе данных, что увеличивает скорость ее работы. Такая архитектура обеспечит

быстрое расширение возможностей системы без необходимости постоянно вносить изменения в клиентские приложения.

2.2.2 Требования к архитектуре серверного приложения

Для разработки серверного приложения следует выбрать шаблон проектирования и использовать его. Выбор верного шаблона проектирования обеспечит приложению гибкость и масштабируемость. Для реализации серверного приложения существует несколько подходящих шаблонов проектирования. В качестве наиболее распространенных стоит рассмотреть следующие шаблоны проектирования:

- Model-View-Controller (MVC, «Модель-Представление-Контроллер»);
- Model-View-Presenter (MVP, «Модель-Представление-Презентер»);
- Model-View-View Model (MVVM, «Модель-Представление-Модель Представления»).

MVC – схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер таким образом, что модификация каждого компонента может осуществляться независимо [14]. Общая структура паттерна проектирования представлена на рисунке 12.

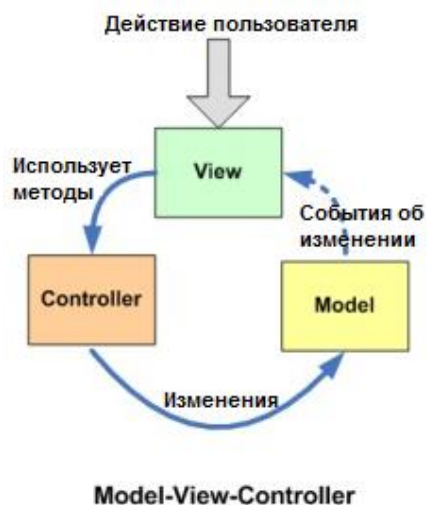


Рисунок 12 – Структура паттерна MVC

MVP – шаблон проектирования, который был разработан для облегчения автоматического модульного тестирования и улучшения разделения ответственности в презентационные отделения логики от отображения. Общая структурная схема паттерна представлена на рисунке 13.

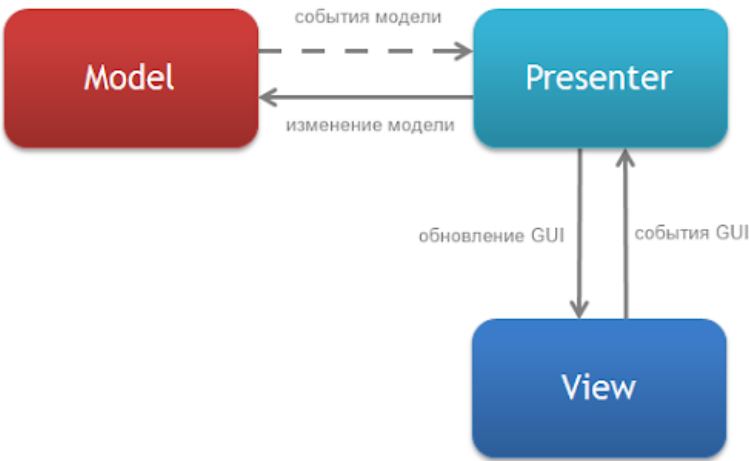


Рисунок 13 – Структура паттерна MVP

Данный подход позволяет создавать абстракцию представления. Для этого необходимо выделить интерфейс представления с определенным набором свойств и методов. Презентер, в свою очередь, получает ссылку на реализацию интерфейса, подписывается на события представления и по запросу изменяет модель.

Паттерн MVVM позволяет отделить логику приложения от визуальной части (представления). Данный паттерн является архитектурным, то есть он задает общую архитектуру приложения. Главным отличием от MVP и MVC является наличие дополнительного компонента ViewModel, отвечающий за связывание и синхронизацию данных между Model и View. Общая структурная схема паттерна представлена на рисунке 14.

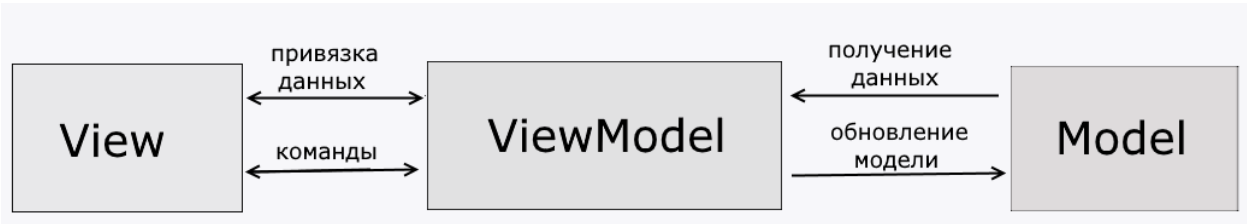


Рисунок 14 – Структура паттерна MVVM

Итогом применения паттерна MVVM является функциональное разделение приложения на три компонента, которые проще разрабатывать и тестировать, а также в дальнейшем модифицировать и поддерживать [15].

Для реализации серверного приложения web-магазина будет использован паттерн MVC, как наиболее подходящий для решения поставленной задачи.

2.3 Проектирование логической схемы базы данных

Для создания схемы базы данных используется интегрированное средство разработки MySQL Workbench, позволяющее создавать логическую модель базы данных, физическую, а также генерировать SQL-код для целевой СУБД [16]. Данное ПО также позволяет составлять отчеты по созданной схеме базы данных. На рисунке 15 представлен результат проектирования логической схемы базы данных.

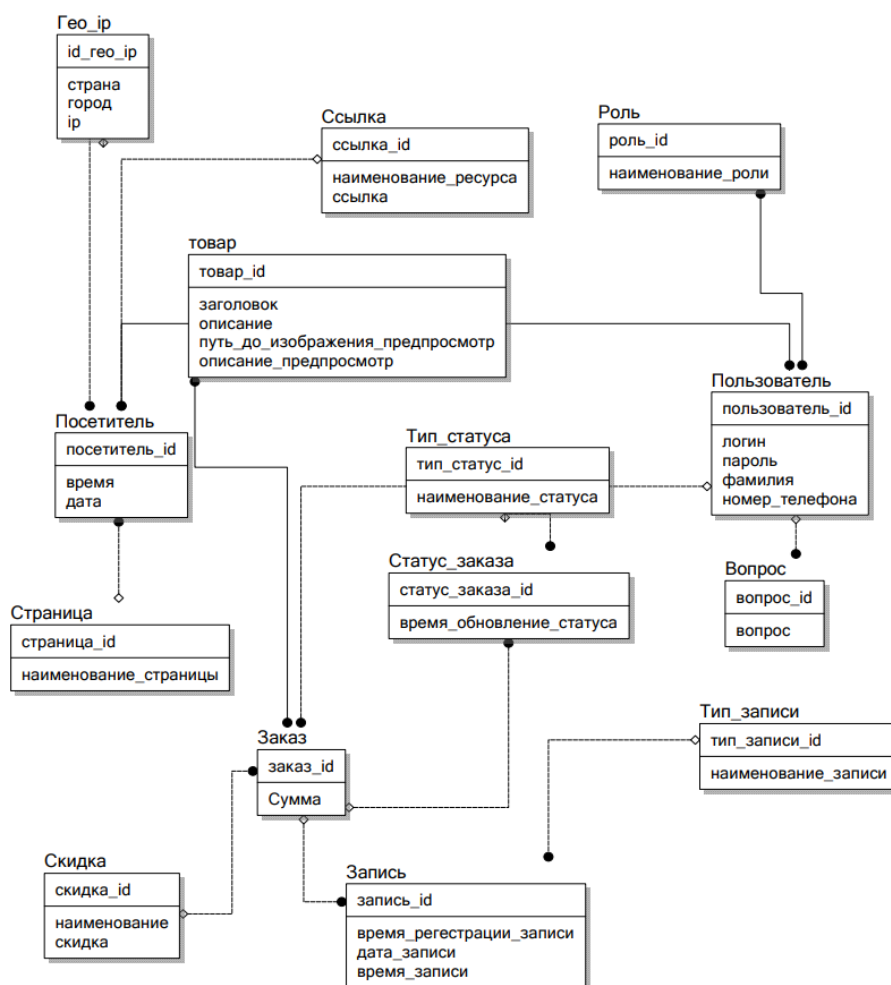


Рисунок 15 – Логическая схема базы данных

Данная схема позволит эффективно хранить и осуществлять доступ к данным. Краткое описание сущностей представлено в таблице 6.

Таблица 6 – Описание сущностей

Название сущностей	Описание
1	2
Гео-Ір	Хранится информация о местоположения клиента, который зашел на сайт
Ссылка	Хранится информация о ссылке с который перешел клиента на страницу сайта
Посетитель	Хранится информация о времени перехода по ссылке клиента на страницу web-магазина
Страница	Хранится информация о странице web-магазина на которую перешел клиента
Товар	Хранится информация о продаваемом товаре и его описание
Скидка	Хранится информация о скидке предоставляемой на товар
Заказ	Хранится информация о сумме заказа и виде товара
Запись	Хранит информацию о времени и заказа и его статусе
Статус заказа	Хранится информация о изменении статуса для конкретного заказа
Тип статуса	Хранится виды статусов для заказа
Тип записи	Хранится информация о типе доставке
Пользователь	Хранится информация о зарегистрированном пользователе
Роль	Хранится информация о роле пользователя зарегистрированного в системе
Вопрос	Хранится информация о вопросе задаваемом пользователем

Описанная структура позволяет хранить информацию о всех бизнес-процессах, происходящих в интернет магазине, а также производить оперативный анализ данных.

2.4 Обоснование выбора СУБД

Для реализации хранения данных пользователя следует выбрать модель данных. В качестве модели для разрабатываемой системы выбран реляционный тип баз данных. В сегменте реляционные баз данных представлено большое количество продуктов. Следует выбрать наиболее

подходящий систему управления базами данных. исходя из требований к решаемой задаче.

Рассмотрим наиболее распространенные продукты, представленные на рынке реляционных систем управления базами данных (СУБД). Для сравнения выбрано следующие ПО:

- MySQL;
- Oracle;
- Microsoft SQL Server;
- PostgreSQL.

Результаты проведения сравнительного анализа по выбранным критериям представлены в таблице 7.

Таблица 7 – Сравнение характеристик баз данных

Характеристика	MySQL	Oracle	Microsoft SQL Server	PostgreSQL
Лицензия	Свободно распространяемая	Платная	Платная	Свободно распространяемая
Максимальный объем хранимой информации	64 ТБ	8 ЭБ	524 272 ТБ	Нет
Качество технической документации	Высокое	Хорошее	Среднее	Хорошее

Для реализации системы будет использоваться СУБД MySQL как наиболее подходящая по характеристикам для решения поставленной задачи.

2.5 Обоснование выбора средств разработки

2.5.1 Обоснование выбора языковых средств разработки

На сегодняшний день на рынке распространено большое количество разнообразных языков программирования, предназначенных для разных целей.

Для реализации серверной части web-приложения наиболее распространены языки Java, C#, Go, Python. Для сравнительного анализа были выбраны следующие языки программирования: Java, C#, Python.

Java – строго типизированный интерпретируемый объектно-ориентированный язык программирования общего назначения. На сегодняшний день торговую марку принадлежат корпорации Oracle. Приложения на Java транслируются в специальный байт-код, что предоставляет возможность исполнять его на разных компьютерных архитектурах и операционных системах. Особенностью языка Java является система безопасности, которая контролирует исполнение программы в рамках виртуальной машины. Некоторые платформы предлагают аппаратную поддержку выполнения для Java. Например, некоторые процессоры ARM-архитектуре поддерживают аппаратное выполнение кода через опцию Jazelle. Для реализации web-приложений на языке Java существует большое количество бесплатно распространяемых фреймворков и библиотек, позволяющих эффективно работать с многопоточностью и обработкой HTTP-запросов [17].

Одним из языков принявших часть концепций Java является язык C#. C# – объектно-ориентированный язык программирования разработанный компанией Microsoft. C# разрабатывался как язык программирования прикладного уровня для исполняемой среды CLR (Common Language Infrastructure, спецификация общезыковой инфраструктуры). Программы C# выполняются в .NET, виртуальной системой выполнения, вызывающей CLR и набор библиотек классов. Исходный код, написанный на языке C# компилируется в промежуточный язык, который соответствует спецификациям CLI. Код на промежуточном языке и ресурсы, в том числе растровые изображения и строки, сохраняются в сборке, обычно с расширением .dll. Сборка содержит манифест с информацией о типах, версии, языке и региональных параметрах для этой сборки. Для реализации web-приложений на языке C# используется платформа разработки ASP.NET, позволяющую реализовывать высокопроизводительные серверные приложения. Для взаимодействия с БД используется технология ADO.NET,

осуществляющая управление подключениями к БД, исполнением запросов и возвращением результатов [18, 19].

Другим языком, позволяющим реализовывать серверные приложения является Python. Python – высокоуровневый язык программирования с динамической типизацией и автоматическим управлением памятью. Язык спроектирован с акцентом на читабельность и лаконичность исходного кода. Python является мультипарадигмальным языком программирования в основе которого лежат: процедурное, структурное, императивное, объектно-ориентированное программирование и функциональное программирование. Язык является интерпретируемым, в качестве стандартного интерпретатора используется свободно распространяемая платформа CPython. Стандартная библиотека Python включает большой набор полезных переносимых функций, начиная от функционала для работы с текстом и заканчивая средствами для написания сетевых приложений. Одной из синтаксических особенностей языка является выделение блоков кода с помощью отступов (пробелов или табуляций), поэтому в Python отсутствуют операторные скобки как в Си подобных языках программирования. Для реализации web-приложений на Python используется несколько распространенных фреймворков, таких как: Django, Pyramid, TurboGears, Flask. Все представленные средства позволяют реализовать серверное приложение любой сложности [20].

Наибольшую гибкость, скорость работы, и читабельность кода в многопоточных высоконагруженных приложениях предоставляет Java за счет использования механизма `CompletableFuture`. Языки C# и Python позволяют работать с многопоточностью, но не так эффективно, как Java [21].

Во всех языках реализована автоматическая очистка и выделения памяти. Наибольшую производительность в выделении памяти и ее использовании предоставляет механизмы виртуальной машины Java, что делает приложение мене ресурсоемким и наиболее производительным. Рассмотрев особенности всех языков программирования для разработки серверного приложения был выбран язык Java.

2.5.2 Обоснование выбора фреймворков и библиотек

Для реализации клиентской части web-приложений, отображаемой в браузере у пользователя в современном мире, существует множество фреймворков и библиотек. Такие фреймворке реализуют синхронизацию данных компонентов, выполняют отображение страниц сайта и делают его адаптивным. Все фреймворке основаны на использование языка JavaScript, html-разметке и CSS (Cascading Style Sheets, Язык стилей). Такой подход позволяет реализовать любую требуемую функциональность. Наиболее распространенные фреймворке для реализации клиентской части web-приложения являются: React.js, Angular, Vue.js, Ember.js. Для сравнения выбраны React.js и Angular как наиболее популярные и производительные среди представленных.

React разрабатывается и поддерживается Facebook, Instagram и сообществом отдельных разработчиков и корпораций. React может использоваться для разработки одностраничных и мобильных приложений. Его цель – предоставить высокую скорость, простоту и масштабируемость. Для простоты создания интерфейсов фреймворк представляет встроенный язык разметки JSX (JavaScript XML). JSX – расширение над стандартным JavaScript синтаксисом, который соединяет в себе возможности html-разметки и CSS, делая разработку более структурированной и легкой. Приложение при этом делится на отдельные JSX-компоненты, которые можно неоднократно переиспользовать в приложении. В фреймворке реализована концепция виртуального DOM (Document Object Model, объектная модель документа). React создает кэш-структуру в памяти, что позволяет вычислять разницу между предыдущим и текущим состояниями интерфейса для оптимального обновления DOM браузера [22, 23].

AngularJS – JavaScript-фреймворк с открытым исходным кодом. Фреймворк работает с HTML, содержащим дополнительные пользовательские атрибуты, которые описываются директивами, и связывает ввод или вывод области страницы с моделью, представляющей собой обычные переменные

JavaScript. Значения этих переменных задаются вручную или извлекаются из статических или динамических JSON-данных. Angular придерживается MVC-шаблона проектирования. Фреймворк расширяет html-разметку и дает возможность обеспечить двустороннюю привязку данных, что уменьшает роль DOM-манипуляций и увеличивает производительность [24].

React обеспечивает большую производительность чем AngularJS за счет виртуализации DOM. При этом React позволяет сделать разработку модульной и масштабируемой за счет переиспользования готовых React-компонентов. Главным минусом React является отсутствие строгого архитектурного паттерна при проектировании ПО, в то время как AngularJS предоставляет возможность адаптировать паттерн MVC.

Для разработки клиентской части web-приложения был выбран фреймворк React, как наиболее подходящий для решение поставленной задачи.

Для разработки серверной части на Java и эффективной разработки используются специальные web-фреймворки. Такие программные решения используются как крупными компаниями, так и маленькими организациями. Наиболее распространенные решения, позволяющие обрабатывать HTTP-запросы и работать в многопоточном режиме являются следующие наиболее распространенные фреймворки: Spring Framework, Blade.

Spring Framework – универсальный фреймворк с открытым исходным кодом для Java-платформы. Spring применим в любом Java-приложении, также существует множество расширений и усовершенствований для построения веб-приложений на Java Enterprise платформе. Spring может быть рассмотрен как коллекция меньших фреймворков или фреймворков во фреймворке. Большинство этих фреймворков может работать независимо друг от друга, однако они обеспечивают большую функциональность при совместном их использовании. Для обеспечения работы приложения web-магазина стоит рассмотреть следующие ключевые части Spring Framework. Центральной частью Spring является контейнер Inversion of Control (IoC, инверсия

управления), который предоставляет средства конфигурирования и управления объектами Java с помощью рефлексии. Контейнер отвечает за управление жизненным циклом объекта: создание объектов, вызов методов инициализации и конфигурирование объектов путем связывания их между собой. Spring MVC является фреймворком, ориентированным на запросы. В нем определены интерфейсы для всех функций современной запросно-ориентированной системы. Цель каждого интерфейса – быть простым и ясным. Все интерфейсы тесно связаны с Servlet API. Фреймворк аутентификации и авторизации – инструмент, позволяющий обеспечить аутентификации, авторизации и управление привилегиями для любого Java-приложения [25].

Blade – это простой MVC-фреймворк на базе Java. Фреймворк позволяет создавать web-приложения и обрабатывать HTTP-запросы [26]. Blade является небольшим и производительным решением, скомпилированный jar-файл с фреймворком весит не более 500 кб. Стандартные меры безопасности в Blade реализованы посредством межсайтовой подделки запросов (CSRF) и защиты XSS. К преимуществам Blade относят:

- поддержка ресурсов web-jar и расширения плагинов;
- модульность;
- производительность;
- малый вес;
- поддержка JDK 1.6.

Spring Framework позволяет реализовать основные функции для web-магазина. Фреймворк предоставляет возможность интегрировать аутентификацию, что позволит ускорить разработку системы. Поддержка Spring. фреймворком IoC и шаблона проектирования MVC, позволит создать гибкое и масштабируемое приложение. Blade предоставляет возможность создать MVC-приложение, позволяющее обрабатывать HTTP-запросы, но остальные преимущества данного ПО не будут в полной мере реализованы при

интеграции в систему web-магазина. В качестве основного фреймворка при разработке серверного приложения был выбран Spring Framework.

2.5.3 Обоснование выбора инструментальных средств

Для разработки web-магазина требуется несколько инструментальных средств. Для разработки серверной части с использованием языка программирования Java и Spring Framework выбрана среда IntelliJ IDEA Community. Бесплатная версия среды позволяет создавать, компилировать и запускать Java-приложения. Отличительной особенностью этого средства является многофункциональная и удобная система рефакторинга кода. Для увеличения производительности труда программиста IntelliJ IDEA предоставляет систему подсказок по набору кода программы. Данное инструментальное средство может использоваться также для создания приложений на языке Kotlin и создания проектов для операционной системы Android [27]. Дополнительной функциональной особенностью среды является возможность отслеживать использование оперативной памяти и загрузки центрального процессора. Помимо работы с Java-приложениями инструментальное средство позволяет подключаться к базам данных, выполнять SQL-запросы и обозревать схему базы данных.

Для разработки клиентской части сайта магазина, выбрано интегрированное средство разработки WebStorm от компании Jet Brains. Средство позволяет создавать JavaScript-приложения с интеллектуальной поддержкой при написании кода. В инструментальное средство также интегрирован сборщик проектов для самых популярных web-фреймворков, таких как: Angular, Vue.js, ReactJs, JQuery [28].

Описанные возможности инструментальных средств лишь часть предоставляемых этим ПО преимуществ. Данные средства развиваются и поддерживаются для повышения эффективности разработки. Именно поэтому для создания системы была выбрана среды WebStorm и IntelliJ IDEA.

3 Разработка web-магазина с анализом конверсий страниц сайта

3.1 Разработка физической схемы базы данных

Для реализации реляционной базы данных в СУБД MySQL создана физическая модель базы данных. На рисунке 16 представлена физическая модель базы данных.

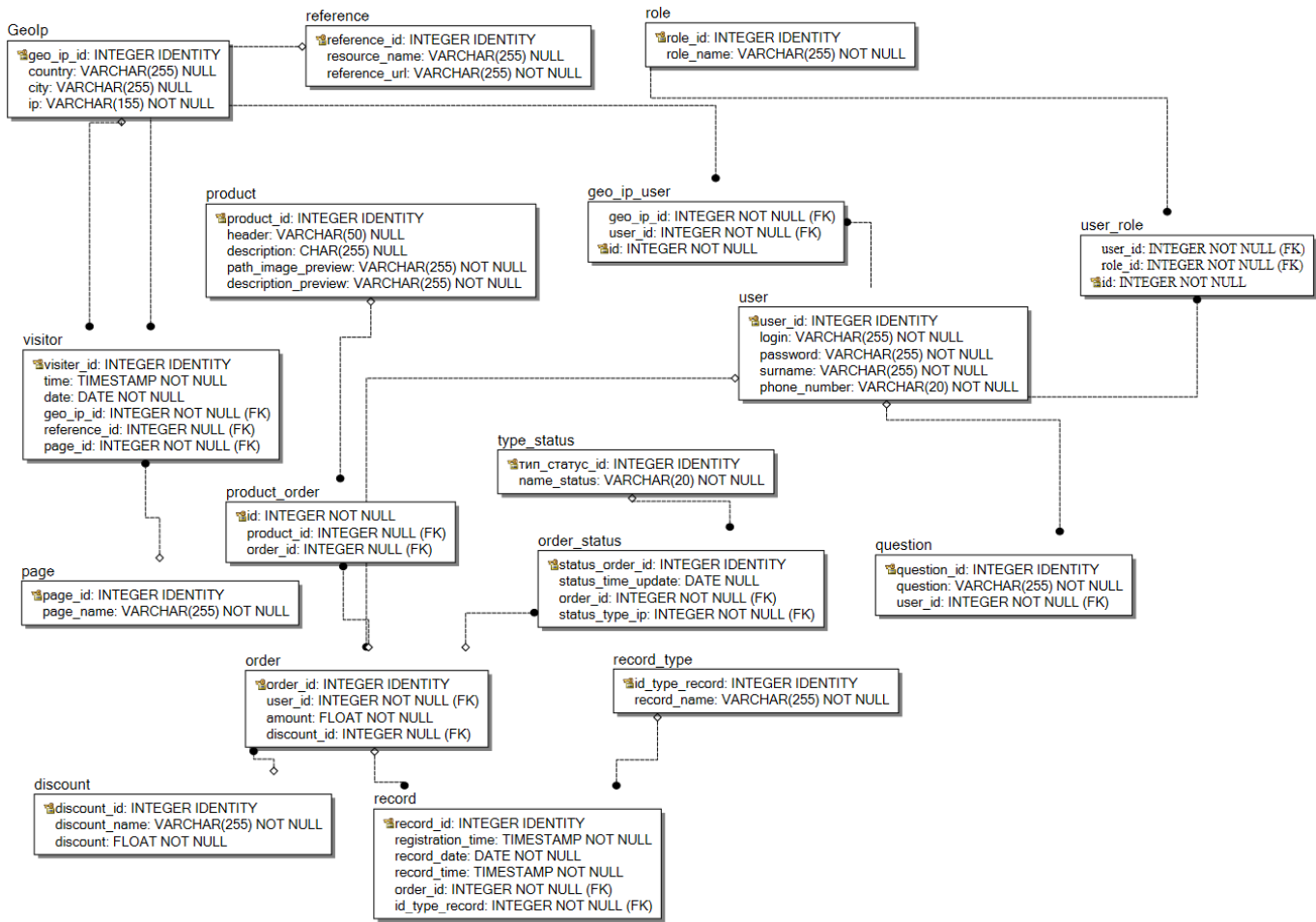


Рисунок 16 – Физическая схема базы данных

В физической модели базы данных связи многие-ко-многим не реализуемы напрямую. Для ее реализации требуется создание промежуточной таблицы. В системе реализованы следующие связи между таблицами. Связи между сущностями представлены в таблице 8.

Таблица 8 – Связи между сущностями

Таблица 1	Таблица 2	Связь
1	2	3
GeoIp	Geo_Ip_user	Один-ко-многим
GeoIp	Visitor	Один-ко-одному

Продолжение таблицы 8

1	2	3
Page	Visitor	Один-ко-многим
Reference	Visitor	Один-ко-многим
Product	Product_order	Один-ко-многим
Order	Product_order	Один-ко-многим
Discount	Order	Один-ко-многим
Order	Record	Один-ко-многим
Record_type	Record	Один-ко-многим
Order_status	Order	Один-ко-многим
Type_status	Order_status	Один-ко-многим
User	Geo_ip_user	Один-ко-многим
User	Question	Один-ко-многим
User	User_role	Один-ко-многим
Role	User_role	Один-ко-многим

Реализация данной схемы базы данных позволит эффективно хранить, извлекать и изменять данные. Для увеличения производительности выполнения запросов «SELECT» на ключевые поля расставлены индексы.

Индекс формируется из значений одного или нескольких столбцов таблицы и указателей на соответствующие строки таблицы и, таким образом, позволяет искать строки, удовлетворяющие критерию поиска. В информационной системе индексы используются в следующих таблицах:

- User;
- Page;
- Order.

Ускорение работы с использованием индексов достигается в первую очередь за счет того, что индекс имеет структуру, оптимизированную под поиск. Для базы данных «MySQL» выбрана структура сбалансированного дерева – «BTREE».

3.2 Разработка структуры серверного приложения

Для наглядного отображения структуры и взаимосвязей в серверном приложении составлена диаграмма классов, описывающую часть основных компоненты программы. Диаграмма классов представлен на рисунке 17.

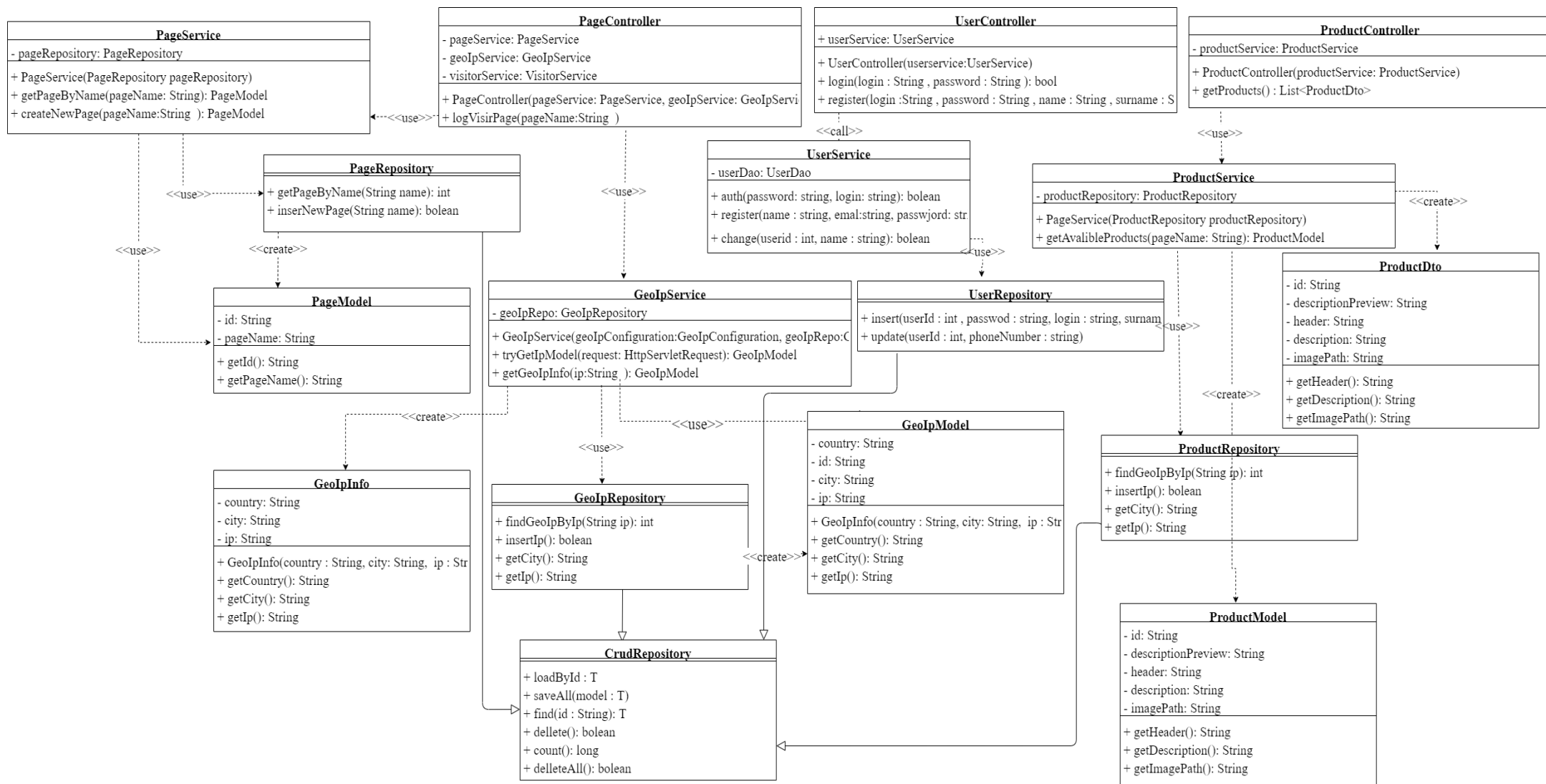


Рисунок 17 – Диаграмма классов

Краткое описание классов представлено в таблице 9.

Таблица 9 – Краткое описание классов

Класс	Описание
PageController	Класс обрабатывает HTTP-запросы связанные с сохранением посещения пользователем страниц сайта
PageService	Класс реализует бизнес логику приложения связанную с сохранением и обработкой полученной статистики со страниц сайта
PageRepository	Класс реализует основные операции для работы с БД и сохранения статистики
PageModel	Класс представляет данные связанные с помещением страниц сайта
GeoIpService	Класс реализует бизнес-логику связанную с идентификацию пользователей на страницах сайтах по его IP
GeoIpInfo	Класс для хранения данных о местоположении по IP из базы mmdb
GeoIpModel	Класс представляет данные связанные с Ip и геолокацией пользователя для сохранения в БД
GeoIpRepository	Класс реализует основные операции для работы с БД и таблицей «GeoIp»
ProductController	Класс обрабатывает HTTP-запросы связанные товарами и заказами пользователями
ProductService	Класс реализует бизнес-логику приложения связанную с заказами и товарами предоставляемые пользователю
ProductDto	Класс представляет данные о товаре для преобразования в json и отправке по сети
ProductModel	Класс представляет данные связанные с товарами и заказами для работы с БД и таблицей «Product»
ProductRepository	Класс реализует основные операции для работы с БД и таблицей «Product»
UserController	Класс обрабатывает HTTP-запросы связанные с доступом к данным пользователя
UserService	Класс реализует бизнес-логику работу с регистрацией и аутентификацией.
UserRepository	Класс реализует методы связанные с доступом к базе данных и таблице «User»
CrudRepository	Абстрактный класс реализует общие методы для выполнения SQL-запросов к БД.

Решение зависимостей между классами и создание объектов через ключевое слово «new» делегируется в приложение на фреймворк Spring IoC.

3.3 Спецификация методов серверного приложения

Фреймворк Spring IoC позволят реализовывать зависимости между классами путем аннотирования класса Java-нотациями: «@Service», «@Repository», «@RestController». В зависимые классы в конструктор по нотации «@Autowired» передаются требуемые объекты [29]. Пример такого объявления представлен в листинге 1 для класса ProductController.

Листинг 1 – Объявление конструктора с зависимыми классами

```
@Autowired
public ProductController(ProductService productService) {
    this.productService = productService;
}
```

Реализация данным образом подстановки зависимостей в нужный класс называется – Dependency injection (DI, внедрение зависимости). Данный метод используется во всех классах проекта.

В классах с постфиксом «Controller» реализуется методы API, предоставляемые для выполнения HTTP-запросов. Методы реализующие основную логику системы представлены в таблице 10.

Таблица 10 – Спецификация методов

№	Название	Назначение	Входные параметры	Возвращаемые значения
1	2	3	4	5
1	getProducts()	Предоставляет данные в виде json о товарах доступных для покупки	-	Цена, полное описание, путь до рисунка, скидка
2	getAvailableProduct	Осуществляет выборку из БД, доступных товаров	-	Цена, краткое описание, полное описание, путь до рисунка, наличие скидки
3	tryGetIpModel	Осуществляет выборку из БД сведений о IP городе и стране, если данных не находится, то такая запись создается	Информация о HTTP запросе	Ip, город, страна, id записи в БД
4	getGeoIpInfo	Осуществляет выборку информации о городе и стране клиента по Ip из mmdb базы	Строка с Ip	Страна, город, Ip

Продолжение таблицы 10

1	2	3	4	5
5	login	Осуществляет проверку существует ли зарегистрированный пользователь в системе	Хэшированный логин и пароль	Токен аутентификации
6	register	Осуществляет регистрацию пользователя в системе	Хэшированный логин и пароль, ФИО пользователя и контактная информация	Токен аутентификации
7	logReference	Принимает HTTP-запросы с клиентской части приложения и осуществляет сохранение статистики	Ссылку с которой перешел пользователь, страница, на которую он перешел, информацию о HTTP-запросе	-
8	logAdReference	Принимает HTTP-запросы с клиентской части приложения и осуществляет сохранение	Ссылку с которой перешел пользователь, страница, на которую он перешел, информацию о HTTP-запросе	-

Описанные в таблице 10 методы находятся в листинге программы, представленном в приложении Б.

4 Руководство пользователя

4.1 Описание интерфейса web-магазина

Программа Web-shop with built in analyzer (WSBA), «система интернет-магазина с возможностью анализа», далее просто «система» имеет дружелюбный User Experience (UX, опыт пользователя) web-интерфейс [30]. Она разработана, чтобы предоставлять клиенту магазину возможность приобрести товар, а бизнес-аналитику произвести анализ экономических показателей.

Главная страница web-магазина представляет собой альбом с доступными для покупки товарами, в правом углу страницы находится кнопка «Вход», позволяющая перейти на страницу логина. Интерфейс главной страницы представлен на рисунке 18.

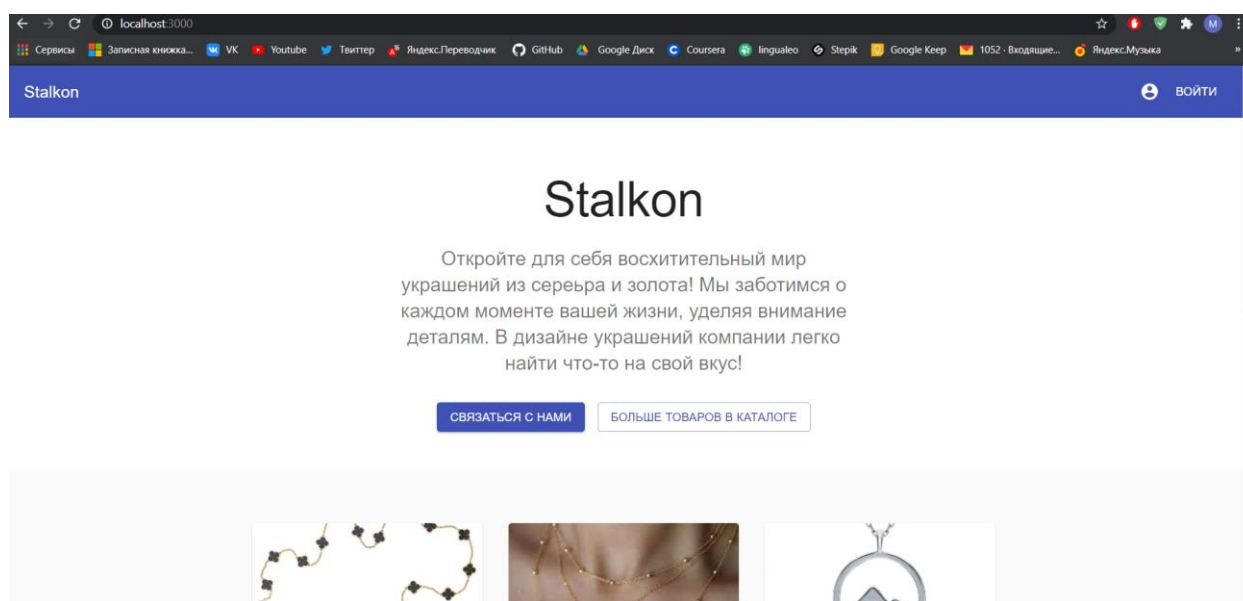


Рисунок 18 – Главная страница приложения

Главная страница сайта представляет собой альбом для просмотра всех доступных товаров. У каждого отображаемый товар можно просмотреть детализированную информацию. Отображение альбома представлено на рисунке 19.

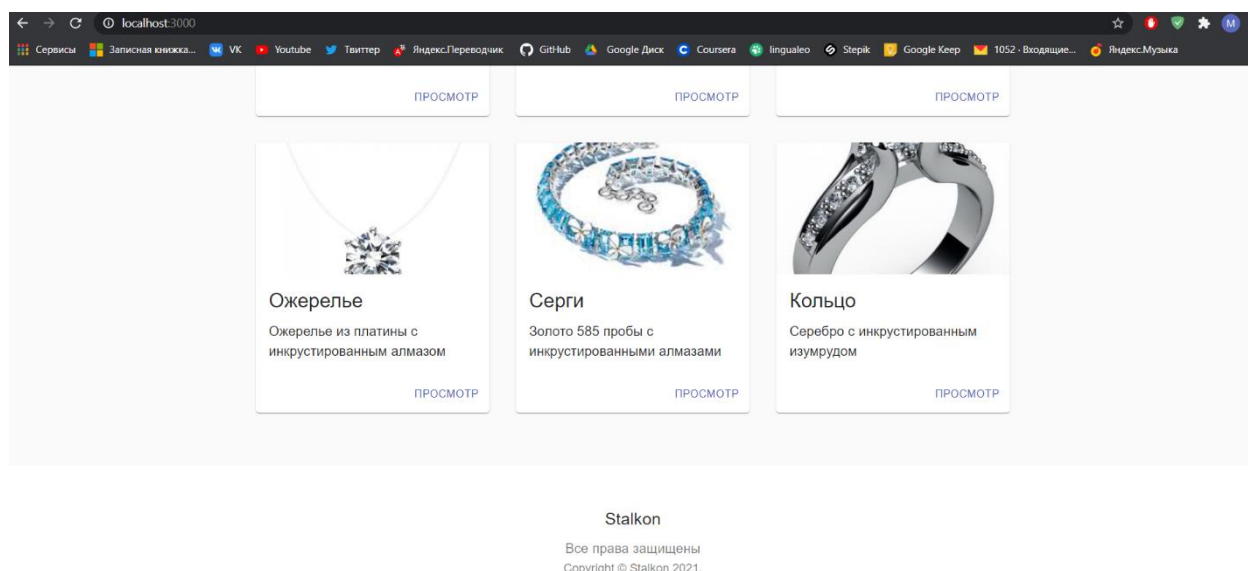


Рисунок 19 – Альбом с доступными товарами

При нажатии на кнопку «Просмотр» на любом товаре, пользователю откроется детализированный просмотр товара. В детализированном просмотре можно осуществить оплату товара, прочитать его описание, выбрать нужные характеристики. На рисунке 20 представлено детализированное описание товара.

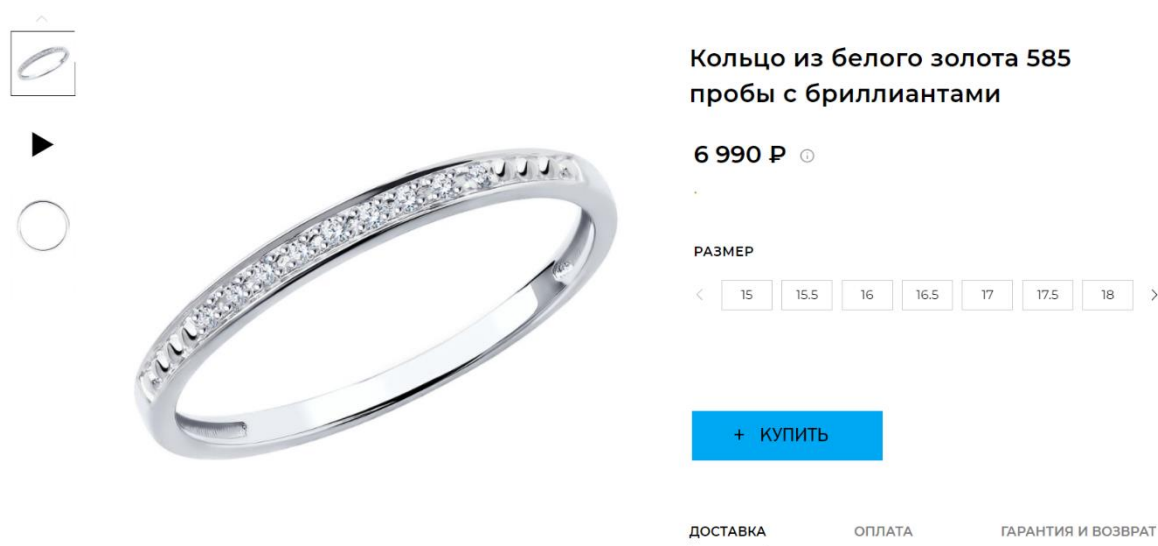


Рисунок 20 – Детализированное описание товара

При нажатии на кнопку «Вход», пользователя перенаправляется на страницу входа. Клиент должен ввести свои логин и пароль. Если

пользователь с введенными данными не находится в системе, клиенту показывается сообщение с ошибкой. Страница входа представлена на рисунке 21.

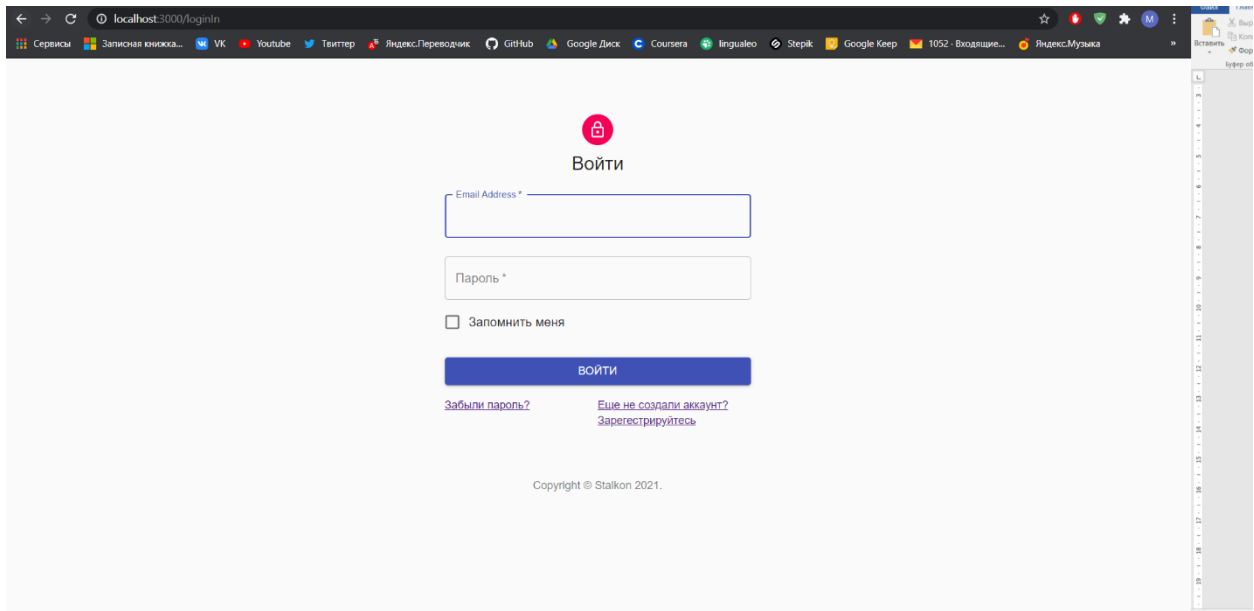


Рисунок 21 – Страница входа

Для создания аккаунта следует перейти по ссылке «Зарегистрироваться». Форма регистрации представлена на рисунке 22.

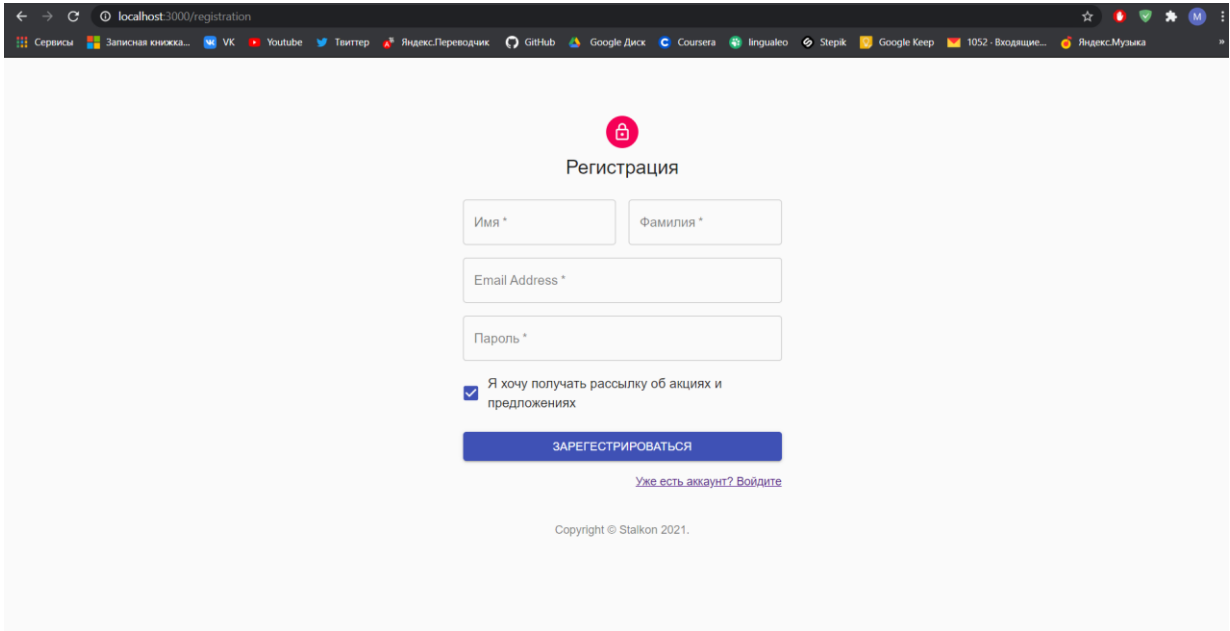
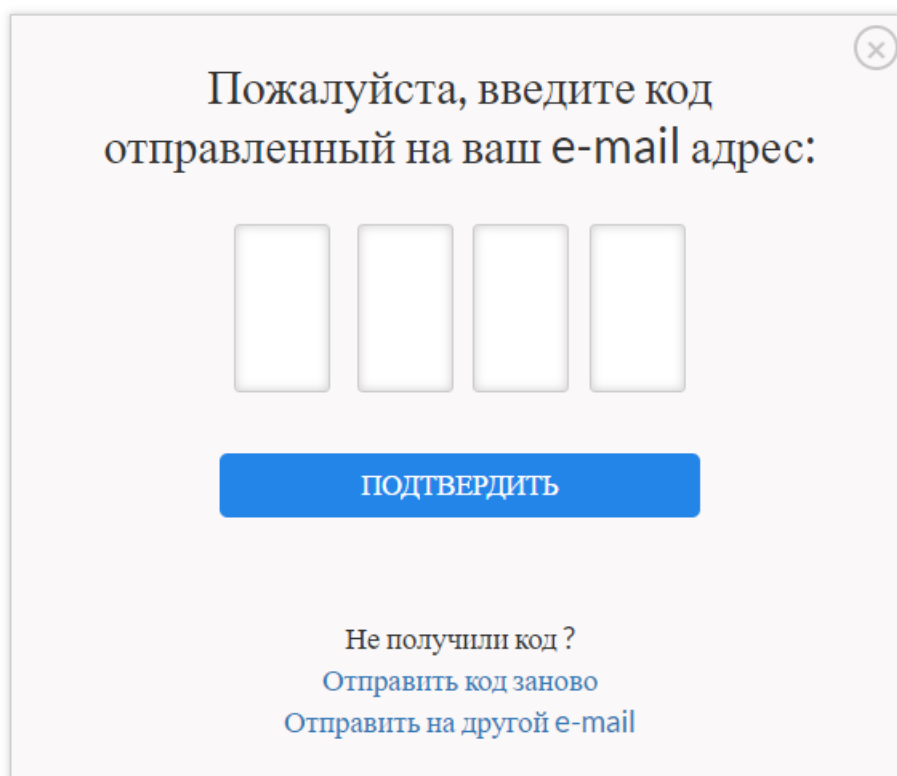


Рисунок 22 – Страница регистрации

Для регистрации пользователь должен ввести свой e-mail адрес, фамилию и имя, а также придумать надежный пароль.

После нажатия на кнопку «Зарегистрироваться» пользователю на e-mail адрес будет отправлено письмо с четырех значным кодом, а в интерфейсе появится диалоговое окно, предлагающее ввести код. Диалоговое окно представлено на рисунке 23.



Пожалуйста, введите код
отправленный на ваш e-mail адрес:

ПОДТВЕРДИТЬ

Не получили код ?
[Отправить код заново](#)
[Отправить на другой e-mail](#)

Рисунок 23 – Окно подтверждения

Из диалогового окна можно отправить код повторно или изменить e-mail для отправки кода.

4.2 Описание интерфейса бизнес-аналитика

Главной страницей приложения бизнес аналитика состоит из набора панелей для проведения анализа по выбранным критериям. Интерфейс разбит на следующие составные части:

- местоположение;
- конверсии;
- прибыль;
- реклама;

— события.

На странице «Местоположение» можно просмотреть статистику по гео-позиции посетителей сайта и активности по конкретно выбранному региону. На рисунке 24 представлена статистика по посещениям сайта и гео-позиции.

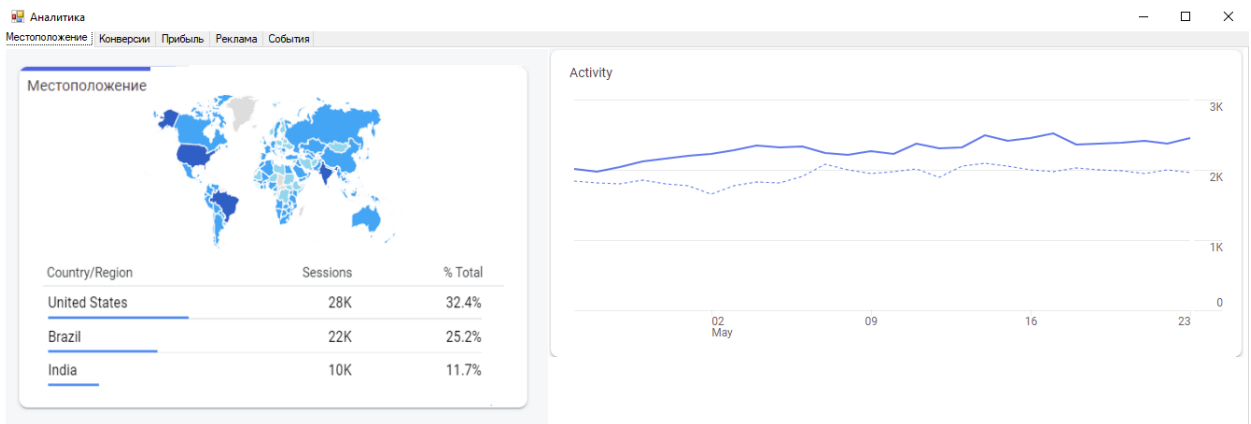


Рисунок 24 – Вкладка «Местоположение»

Для просмотра статистики по событиям с конверсиями в программе представлена вкладка «Конверсии». На рисунке 25 представлены отправленные с сайта события. Для каждого события представлена динамика роста или убыли в численном и процентном соотношении. События, отображаемые на вкладке связаны с целевыми действиями пользователя.

События с конверсиями				
Название события ↑	Кол-во	Изменение %	Кол-во	Изменение %
add_to_cart	0	0%	0	0%
add_to_wishlist	0	0%	0	0%
app_store_subscription_convert	0	0%	0	0%
app_store_subscription_renew	0	0%	0	0%
app_update	494	↓ 6.6%	-	-
begin_checkout	0	0%	0	0%
completed_5_levels	2,319	↑ 73.7%	371	↑ 18.5%
ecommerce_purchase	0	0%	0	0%
first_open	23,303	↑ 34.4%	-	-
in_app_purchase	4	↓ 77.8%	\$7.61	↓ 89.0%

Рисунок 25 – Вкладка «Конверсии»

Для оценки экономической эффективности и прибыли, бизнес-аналитик может воспользоваться вкладкой «Прибыль». На данной вкладке отображены

основные экономические показатели. Пользователь может просмотреть статистику по новым привлеченным пользователям в выбранный промежуток времени, а также о количестве новых посещений за выбранный промежуток.

Для определения наиболее прибыльного товара пользователю предоставляется таблица со статистикой по совершенным в интернет-магазине покупкам. Также для оценки заинтересованности пользователей тем или иным товарам приложение позволяет просмотреть количество времени, проведенного на целевой странице. На рисунке 26 представлен вкладка «Прибыль».

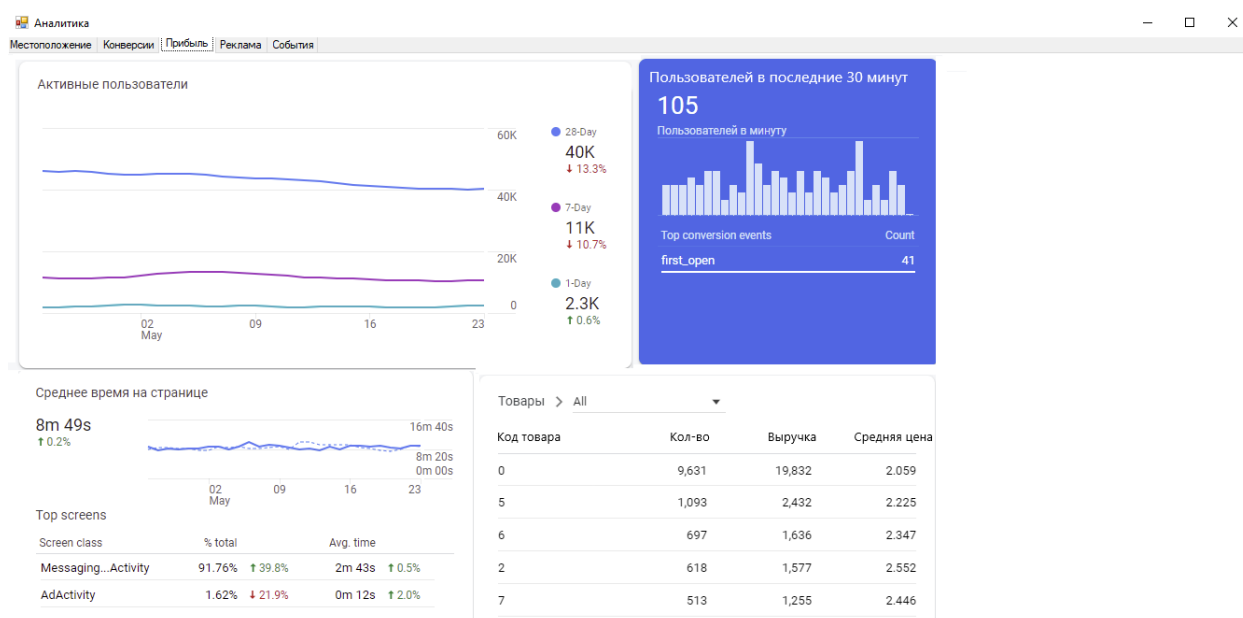


Рисунок 26 – Вкладка «Прибыль»

Для просмотра сведений о эффективности рекламной политики предназначена вкладка «Реклама». На ней отображено количество средств, затраченных на рекламу, а также динамика изменения продаж за выбранной промежуток времени.

Используемые рекламные интеграторы отображаются в таблице и позволяют просмотреть сайты с которых был осуществлен переход на страницу магазина. На вкладке «Реклама» представленной на рисунке 27 отображаются сведения о подключенном рекламном сервисе «AdWords» и текущем статусе.

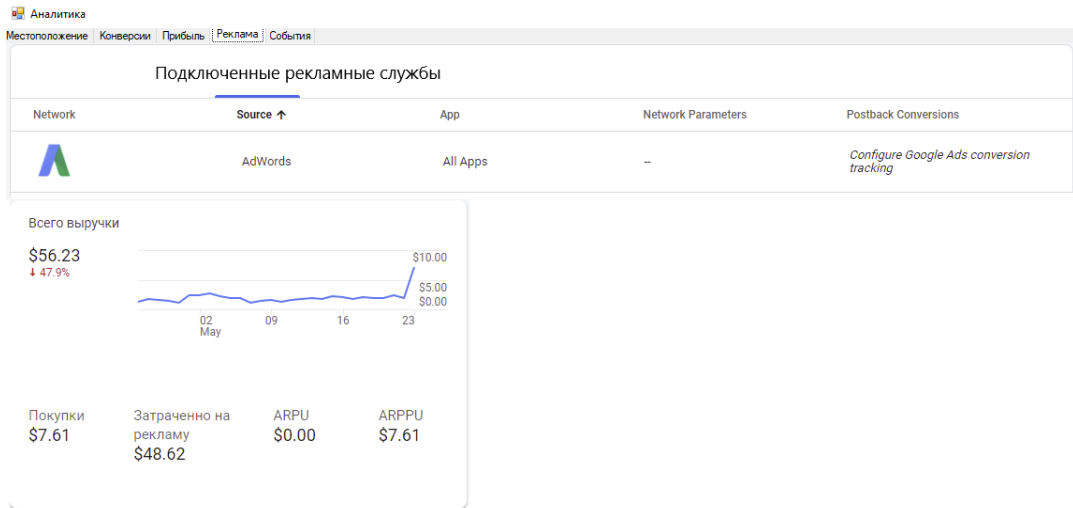


Рисунок 27 – Вкладка «Реклама»

Для принятия управленческих решений зачастую требуются дополнительные метрики, позволяющие определить успешность продвижения товара. Для этих целей применяется дополнительные события, отправляемые с сайта интернет-магазина. На вкладке «События» можно просмотреть количественные и процентные метрики для интернет-магазина. Пример отображения всех доступных дополнительных событий представлен на рисунке 28.

Существующие события

Название события ↓	Количество	% изменений	Пользователи	% изменение
use_extra_steps	15,699	↑ 186.4%	3,218	↑ 66.8%
spend_virtual_currency	16,225	↑ 173.2%	3,540	↑ 63.0%
session_start	86,220	↑ 17.6%	33,289	↑ 28.5%
select_content	122,727	↑ 22.5%	25,818	↑ 34.2%
screen_view	1,912,078	↑ 23.4%	32,154	↑ 29.6%
post_score	170,802	↑ 24.8%	14,551	↑ 48.3%
page_view	1,464	↑ 2.5%	1,403	↑ 1.8%
os_update	1,978	↓ 33.2%	1,861	↓ 34.3%
no_more_extra_steps	8,894	↑ 166.5%	2,987	↑ 66.7%

Рисунок 28 – Вкладка «События»

По каждому представленному событию можно просмотреть детализацию нажав на интересующую строку. После чего откроется диалоговое окно, представленное на рисунках 29.

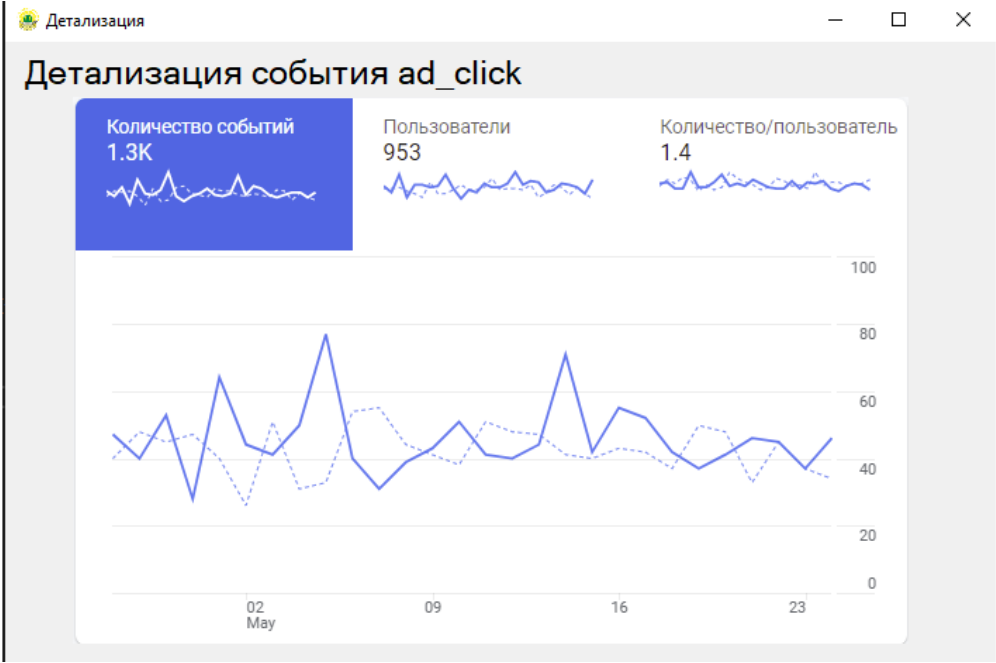


Рисунок 29 – Детализация статистики по событию «ad_click»

Информация о событии доступна в нескольких разрезах. На рисунке 30 представлена информация о событии «ad_click» в разрезе количества пользователей, совершивших целевое действие на странице web-магазина.

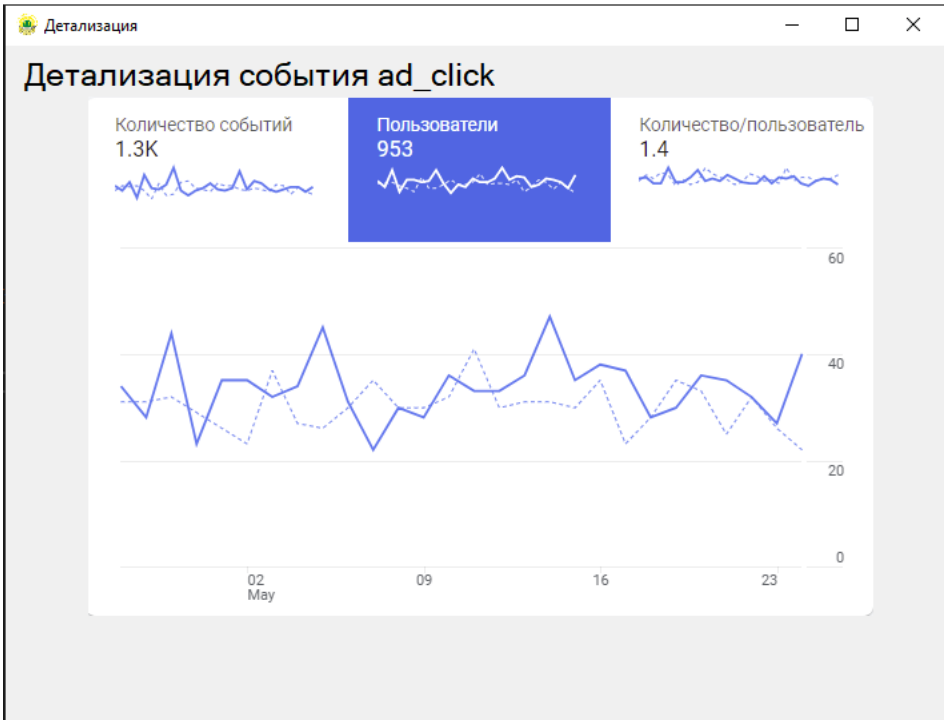


Рисунок 30 – Детализация статистики по событию «ad_click» в разрезе количества пользователей

Для просмотра детализации отношения количества совершенных целевых действий к общему числу пользователей создан отдельный фильтр для отображения данных (рисунок 31).

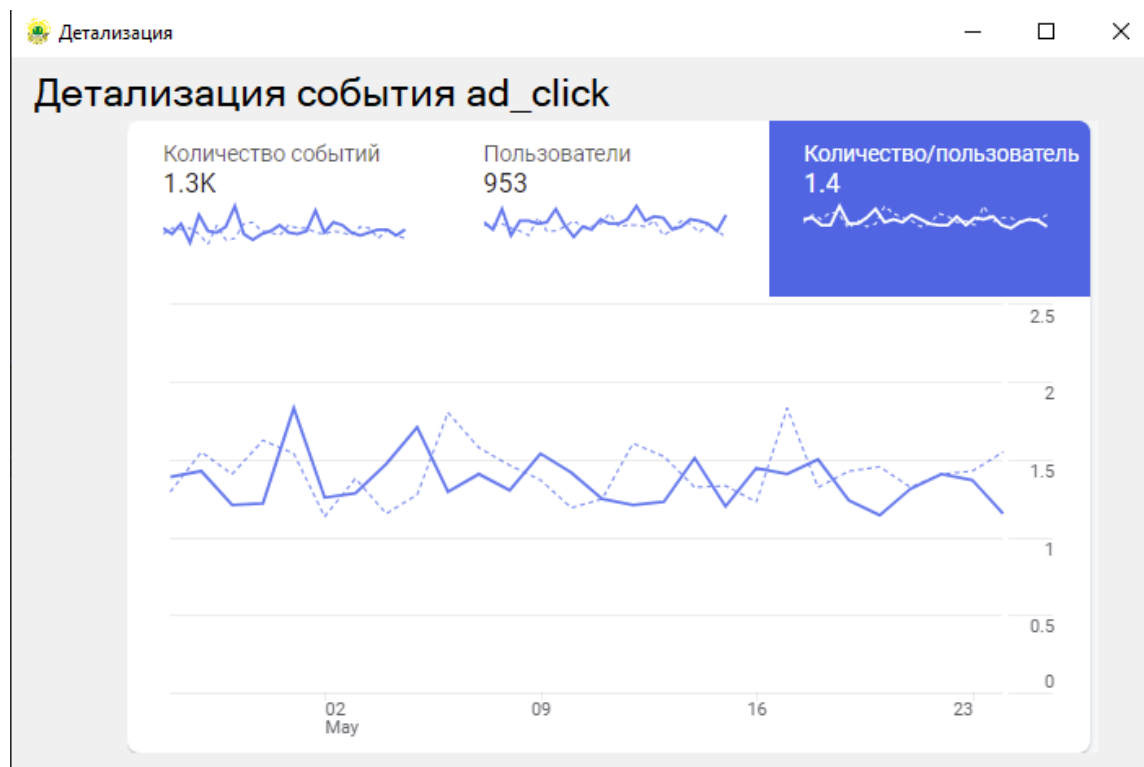


Рисунок 31 – Детализация статистики по событию «ad_click»

По мере совершенствования системы, справочная информация и руководство пользователя будут обновляться.

5 Тестирование приложения

5.1 Выбор методологии тестирования ПО

Под тестированием программных средств понимается процесс исследования ПО с целью проверки соответствия между реальным поведением программы и ожидаемым поведением [31].

Современные методологии тестирования программных средств не позволяют выявить все ошибки и однозначно сказать, что ПО не имеет встроенных дефектов.

В зависимости от поставленных целей выделяют разные виды тестирования. Можно выделить следующие наиболее общие виды тестирования:

- функциональные;
- нефункциональные;
- связанные с изменениями.

Функциональное тестирование направлено на проверку реализуемости функциональных требований. Функциональные тесты могут проводиться на всех уровнях тестирования. В рамках этого вида тестирования проводится имитация фактического использования системы, т.е. функциональные тестирование рассматривают внешнее поведение системы.

В отличии от функционального тестирования, Нефункциональное направлено на проверку реализуемости нефункциональных требований. К таким требованиям обычно относят следующие:

- производительность;
- удобство пользования;
- надежность.

В рамках нефункционального тестирования проверяются глобальные свойства системы, а не соответствие требованиям заказчикам.

На каждом этапе разработке ПО важна проверка работоспособности системы. Для выполнение этой задачи применяют связанное с изменениями

тестирование. Оно зачастую проводится после устранения дефекта или ошибки в ПО для подтверждения того факта, что проблема была действительно решена.

Для проведения тестирования системы web-магазина выбран метод функционального тестирования системы.

5.2 Тестирование приложения

Функциональное тестирование – процесс, основанный на заранее известном или предполагаемом поведении пользователя, детальном анализе и изучении функциональной спецификации приложения. В таблице 11 представлено тест-кейсы для тестирования системы.

Таблица 11 – Описание тестирования

Дымовое тестирование	Тестирование критического пути	Расширенное тестирование	Результат тестирования критического пути
1	2	3	4
Зарегистрироваться	На странице регистрации ввести e-mail и новый пароль. Ввести код подтверждения из e-mail письма.	Переход на страницу регистрации с главной страницы сайта. Переход на страницу регистрации через прямую ссылку. Вводит пароль и e-mail в форме для регистрации. Ввод пароль и e-mail не удовлетворяющим требованиям. Ввод кода подтверждения из e-mail. Ввод неправильного кода.	Рисунки В.1, В.2
Аутентифицироваться	На странице входа ввести логин и пароль.	Прейти на страницу входа. Ввести логин и пароль в форме входа. Сделать запрос для получения токена-аутентификации через строку запроса.	Рисунки В.3, В.4

Продолжение таблицы 11

1	2	3	4
Просмотр главной странице сайта.	На главной странице отображаются доступные товары. Все товары доступны для взаимодействия.	На главную страницу можно перейти из строки браузера. Элементы страницы расположены корректно и доступны для взаимодействия. С главной страницы можно перейти на страницу регистрации и входа в аккаунт. При нажатии на товар открывается страница связанная с этим товаром.	Рисунок В.5
Анализ географической расположенности клиента	На главной странице приложения бизнес-аналитика открыть вкладку «местоположение».	На главной странице приложения бизнес-аналитика открыть вкладку «местоположение» ввести дату за который нужно произвести анализ. Ввести время промежуток времени в который нужно осуществить анализ.	Рисунок В.6
Анализ конверсий страниц сайта.	На главной странице приложения бизнес-аналитика открыть вкладку «конверсии».	На главной странице приложения бизнес-аналитика открыть вкладку «конверсии». Выбрать время за которые нужно отобразить данные. Отсортировать события с конверсиями по всем критериям. Просмотреть таблицу событий.	Рисунок В.7
Анализ прибыли	На главной странице приложения бизнес-аналитика открыть вкладку «конверсии» после чего перейти на вкладку «прибыль».	На главной странице приложения бизнес-аналитика открыть вкладку «конверсии» после чего перейти на вкладку «прибыль». Просмотреть графики с данными и сравнить с числовыми показателями.	Рисунок В.8

Результаты проведения данных тест-кейсов показали корректную и безошибочную работу ИС.

5.3 Модульное тестирование

Модульное тестирование процесс в программировании, позволяющий проверить на корректность отдельные модули исходного кода программы. Метод состоит в том, чтобы писать тесты для каждого нетривиального метода или алгоритма. Отсутствие модульного тестирования при написании кода значительно увеличивает уровень дефектов при дальнейшем интеграционном тестировании.

В информационной системе интернет-магазина с возможностью проведения анализа – модульное тестирование использовалось с целью проверки корректности работы модуля определения местоположения по Ip пользователя. Для проведения тестирования используется библиотека с открытым исходным кодом Junit, позволяющая осуществлять модульное тестирование Java-программ [32]. Для проверки была составлена таблица 12 с тестовыми случаями и ожидаемыми результатами.

Таблицы 12 – Модульное тестирование

№	Тестовые данные	Ожидаемый результат	Полученный
1	217.118.90.224	RU	RU
2	217.118.90.224	Saratov	Saratov
3	127.0.0.1	null	null
4	127.0.0.1	null	null

Среда разработки IntelliJ IDEA позволяет запускать тесты через интерфейс и отображать результаты.

На рисунке 32 представлен результат запуска тестов.

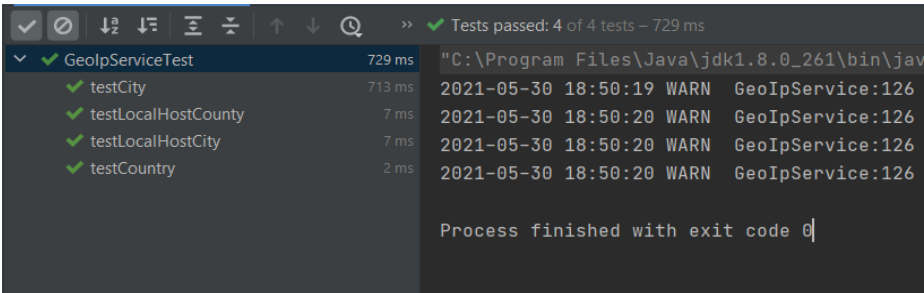


Рисунок 32 – Результат выполнения модульных тестов

В результате проведения тестирования были получены ожидаемые результаты. Дефектов и ошибок в ходе проведения модульного тестирования выявлено не было.

5.4 Метрики кода

Метрикой программного обеспечения называется мера, которая служит для получения численного значения некоторого свойства программного обеспечения или его спецификаций. В работе проведен расчет метрики Холстеда для метода сохраняющего событие перехода с рекламного источника. Тело метода представлено на рисунке 33.

```
@PostMapping("/logReference")
public void logReference(@RequestParam String headerReference,
                        @RequestParam String landingPage,
                        HttpServletRequest request) {
    GeoIpModel geoIpModel = geoIpService.tryGetIpModel(request);
    if (geoIpModel == null) {
        logger.warn("Can't log reference");
        return;
    }

    PageModel pageModel = pageService.getPageByName(landingPage);
    if (pageModel == null) {
        pageModel = pageService.createNewPage(landingPage);
    }

    if (headerReference == null || headerReference.isEmpty()) {
        logger.warn("Reference is null or empty");
        visitorService.logVisitor(geoIpModel.getId(), pageModel.getId(), referenceId: null);
    }

    ReferenceModel referenceModel = referenceService.getReferenceModel(headerReference);
    if (referenceModel == null) {
        logger.warn("Invalid reference");
        return;
    }

    visitorService.logVisitor(geoIpModel.getId(), pageModel.getId(), referenceModel.getId());
}
```

Рисунок 33 – Тело метода, реализующего сохранение рекламной ссылки

Для проведения расчетов метрики необходимо выяснить значение таких параметров, как количество уникальных операторов, количество уникальных операндов, общее количество операторов, общее количество операндов. Полученные результаты находятся в таблице 13.

Таблица 13 – Операторы и операнды

Операторы	Количество	Операнды	Количество
1	2	3	4
if()	4	pageModel	4
=	4	headerReference	3

Продолжение таблицы 13

1	2	3	4
==	4	landingPage	2
	1	request	1
		geoIpModel	3
		referenceModel	2
n1 = 4	N1 = 13	n2 = 3	N2 = 15

На основе полученных показателей определяем словарь программы по формуле (1):

$$n = n1 + n2 = 7, \quad (1)$$

Также определяем длину подпрограммы по формуле (2):

$$N = N1 + N2 = 27. \quad (2)$$

При наличии этих значений можно рассчитать объем кода (3) и уровень качества программирования (4):

$$V = N \times \log_2 n, \quad (3)$$

$$L' = \frac{(2 \times n2)}{(n1 \times N2)}. \quad (4)$$

После подстановки значений в формулы, получаем следующие значения: $V = 75,6$; $L' = 0,1$. Теперь посчитаем сложность качества программы, используя вычисленные значения объема кода и уровня качества программирования (5):

$$Ec = \frac{V}{L'} \times 2.$$

В результате получаем: $Ec = 1512$. Данный результат означает, что разработанный код подпрограммы является в достаточной мере компактным и простым для понимания. Это объясняется тем, что в коде присутствуют множество операндов.

Заключение

В ходе выполнения выпускной квалификационной работы был осуществлен обзор предметной области, в процессе которого была рассмотрена актуальность создания системы web-магазина с анализом конверсий сайта. Был проведен обзор существующих программных средств, специализирующихся на создании web-сайтов с возможностью поведения анализа. В рамках выпускной квалификационной работы была поставлена задача на разработку и определен бюджет проекта. В процессе проектирования приложения была проведена разработка требований к информационной системе и к основным ее компонентам. Процесс разработки требований включил в себя определение требований к архитектурам, используемым языкам программирования и инструментальным средствам.

В результате выполнения выпускной квалификационной работы был разработан web-магазин с возможностью анализа конверсий.

Список использованных источников

1. Блэнд Д. Тестирование бизнес-идей / Блэнд Дэвид, Остервальдер Александр. – П.:Альпина Паблишер, 2020 - 356 с.
2. Образовательная платформа WebPromo Experts – Как правильно собирать и использовать данные о действиях пользователей на сайте. [Электронный ресурс]. – URL:<https://webpromoexperts.net/blog/kak-pravilno-sobirat-i-ispolzovat-dannye-o-dejstviyah-polzovatelej-na-sajte/> (дата обращения 18.03.2021).
3. ООО «Торговый портал» – Веб-аналитика для коммерческого сайта [Электронный ресурс]. – URL: <https://beseller.by/blog/web-analitika-dlya-kommercheskogo-sayta/> (дата обращения 20.03.2021).
4. НКО «Ассоциация компаний интернет-торговли» – Эксперты сообщили о росте рынка интернет-торговли в России на 44 процента [Электронный ресурс]. – URL: https://akit.ru/experts_said_forecast/ (дата обращения 21.03.2021).
5. Группа компаний «Современные технологии управления» – Система KPI (Key Performance Indicator): разработка и применение показателей бизнес-процесса. Показатели эффективности [Электронный ресурс]. – URL: https://www.businessstudio.ru/articles/article/sistema_kpi_key_performance_indicator_razrabotka_i/ (дата обращения 22.03.2021).
6. Free Software Foundation inc. – Стандартная общественная лицензия GNU (GPL), версия 2 [Электронный ресурс]. – URL: <https://www.gnu.org/licenses/old-licenses/gpl-2.0.ru.html> (дата обращения 22.03.2021).
7. WordPress.org – WordPress [Электронный ресурс] –URL: <https://ru.wordpress.org/plugins/> (дата обращения 23.03.2021).
8. Alphabet Inc. – Firebase [Электронный ресурс]. – URL: <https://firebase.google.com/?hl=ru> (дата обращения 22.03.2021).
9. ОАО «Tilda Publishing» – Tilda [Электронный ресурс]. – URL: <https://tilda.cc/ru/> (дата обращения 23.03.2021).

10. Google.inc – Material Design [Электронный ресурс]. – URL.: <https://material.io/design> (дата обращения 23.03.2021).
11. Atlassian inc – Что такое диаграмма Ганта [Электронный ресурс]. –URL: <https://www.atlassian.com/ru/agile/project-management/gantt-chart> (дата обращения 23.03.2021).
12. Microsoft Corporation – Выбор подходящего отчета в Project [Электронный ресурс]. – URL: <https://cutt.ly/IiJ6QJt> (дата обращения 20.03.2021)
13. Национальный открытый университет Интуит – Различные архитектурные решения, используемые при реализации многопользовательских СУБД [Электронный ресурс]. - URL: <https://intuit.ru/studies/courses/508/364/lecture/8643?page=2>(дата обращения 25.03.2021).
14. Национальный открытый университет Интуит – Знакомство с подходом MVC [Электронный ресурс]. - URL: <https://intuit.ru/studies/courses/527/383/lecture/8933> (дата обращения 27.03.2021).
15. METANIT - Определение паттерна MVVM [Электронный ресурс]. - URL: <https://metanit.com/sharp/wpf/22.1.php> (дата обращения 01.04.2021).
16. Oracle Corporation – About workbench [Электронный ресурс]. - URL: <https://www.mysql.com/products/workbench/> (дата обращения 23.03.2021).
17. Oracle Corporation – Oracle Java [Электронный ресурс]. - URL: <https://www.oracle.com/ru/java/> (дата обращения 23.03.2021).
18. Microsoft Corporation - Краткий обзор языка C# [Электронный ресурс]. – URL: <https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/> (дата обращения 05.04.2021).
19. Microsoft Corporation – What is ASP.NET? [Электронный ресурс]. – URL: <https://dotnet.microsoft.com/apps/aspnet> (дата обращения 05.04.2021).

20. Python Software Foundation - Python For Beginners [Электронный ресурс]. – URL.: <https://www.python.org/about/apps/> (дата обращения 08.04.2021).
21. Oracle Corporation – CompletableFuture [Электронный ресурс]. – URL.: <https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/CompletableFuture.html> (дата обращения 10.04.2021).
22. Facebook Inc. - Знакомство с React [Электронный ресурс]. – URL.: <https://ru.reactjs.org/> (дата обращения 10.04.2021).
23. MSDN Web Docs – Что такое Объектная Модель Документа (DOM)? [Электронный ресурс]. – https://developer.mozilla.org/ru/docs/Web/API/Document_Object_Model/Introduction (дата обращения 10.04.2021).
24. Angular – Особенности и преимущества [Электронный ресурс]. – URL.: <https://angular.io/features> (дата обращения 20.04.2021).
25. Spring – Введение в Spring [Электронный ресурс]. – URL.: <https://spring.io/why-spring> (дата обращения 25.04.2021).
26. Baeldung – Blade. A Complete Guidebook [Электронный ресурс]. – URL.: <https://www.baeldung.com/blade> (дата обращения 25.04.2021).
27. JetBrains s.r.o. – Возможности IntelliJ IDEA [Электронный ресурс]. – URL.: <https://www.jetbrains.com/ru-ru/idea/> (дата обращения 27.04.2021).
28. JetBrains s.r.o. - Возможности WebStorm [Электронный ресурс]. – URL.: <https://www.jetbrains.com/ru-ru/webstorm/> (дата обращения 01.05.2021).
29. Baeldung – Spring Core Annotations [Электронный ресурс]. – URL.: <https://www.baeldung.com/spring-core-annotations> (дата обращения 01.05.2021).
30. ОАО «Tilda Publishing» - Что такое UX дизайн? [Электронный ресурс]. – URL.: <https://tilda.education/articles-what-is-ux-design> (дата обращения 02.05.2021).
31. Майерс Г. Искусство тестирования программ / Майерс Гленфорд, Баджетт Том. - П.: Вильямс, 2020 – 272с.
32. The JUnit Team - JUnit 5 is the next generation of Junit [Электронный ресурс]. – URL.: <https://junit.org/junit5/> (дата обращения 15.05.2021).

Приложение А
ГЛОССАРИЙ
(справочное)

Электронная коммерция – сфера экономики, которая включает в себя все финансовые и торговые транзакции, осуществляемые при помощи компьютерных сетей, и бизнес-процессы, связанные с проведением таких транзакций.

Информационная система (ИС) – система, предназначенная для хранения, поиска и обработки информации.

Key Performance Indicators (KPI) – это числовые показатели деятельности, которые помогают измерить степень достижения целей или оптимальности процесса, а именно: результативность и эффективность.

Cost Per Action (CPA) – показатель, с помощью которого рекламодатель может рассчитать, во сколько ему обошлось целевое действие пользователя на сайте.

Unified Modeling Language (UML) – язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

Application programming interface (API) – описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой.

Фреймворк – программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

Программные средства (ПС) – это совокупность программ, которые и заставляют аппаратную часть системы выполнять необходимые действия.

Java Virtual Machine (JVM) – основная часть исполняющей системы Java.

Model View Controller (MVC) – схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер.

Model-View-Presenter (MVP) – шаблон проектирования, производный от MVC, который используется в основном для построения пользовательского интерфейса.

Система управления базами данных (СУБД) – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

База данных (БД) – представленная в объективной форме совокупность самостоятельных материалов.

Jazelle (прямое выполнение байт-кода) – это расширение, которое позволяет некоторым процессорам ARM выполнять Байт-код Java в аппаратном обеспечении в качестве третьего состояния выполнения наряду с существующими режимами ARM.

Common Language Runtime (CLR, общезыковая исполняющая среда) – исполняющая среда для байт-кода, в которой компилируются программы, написанные на .NET-совместимых языках программирования. CLR является одним из основных компонентов пакета Microsoft .NET Framework.

Cascading Style Sheets (CSS) – формальный язык описания внешнего вида документа, написанного с использованием языка разметки.

Inversion of Control (IoC) – архитектурное решение, упрощающее расширение возможностей системы, при котором поток управления программы контролируется фреймворком.

Structured Query Language (SQL) – декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

Dependency injection (DI) – процесс предоставления внешней зависимости программному компоненту.

Приложение Б
ЛИСТИНГ ПРОГРАММЫ
(рекомендуемое)

Листинг Б.1 – Код файла MetalConstructionServer.java

```
package com.metal.construction;

import com.metal.construction.core.config.ServerConfiguration;
import org.apache.log4j.Logger;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplication;
import
org.springframework.boot.context.properties.EnableConfigurationP
roperties;
import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.Bean;

@SpringBootApplication
@EnableConfigurationProperties(ServerConfiguration.class)
public class MetalConstructionServer {
    private static final Logger logger =
Logger.getLogger(MetalConstructionServer.class);
    public static void main(String[] args) {
        try {
            SpringApplication.run(MetalConstructionServer.class,
args);

            logger.info("SERVER START SUCCESSFUL");
        } catch (Exception e) {
            logger.error("START ERROR", e);
            System.exit(1);
        }
    }

    @Bean
    public CommandLineRunner
commandLineRunner(ApplicationContext ctx) {
        return args -> {
            System.out.println("Let's inspect the beans provided
by Spring Boot:");
        };
    }
}
```

Листинг Б.2 – Код файла GeoIpConfiguration.java

```
package com.metal.construction.geoIp.config;

import
org.springframework.boot.context.properties.ConfigurationPropert
ies;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.PropertySource;

@Configuration
```

```

@PropertySource("classpath:application.properties")
@ConfigurationProperties(prefix = "geoip")
public class GeoIpConfiguration {
    private String type;
    private String dbPath;
    private int dbCacheSize;
    private boolean fake;
    public boolean isFake() {
        return fake;
    }
    public void setFake(boolean fake) {
        this.fake = fake;
    }
    public String getType() {
        return type;
    }
    public void setType(String type) {
        this.type = type;
    }
    public String getDbPath() {
        return dbPath;
    }
    public void setDbPath(String dbPath) {
        this.dbPath = dbPath;
    }
    public int getDbCacheSize() {
        return dbCacheSize;
    }
    public void setDbCacheSize(int dbCacheSize) {
        this.dbCacheSize = dbCacheSize;
    }
}

```

Листинг Б.3 – Код файла GeoIpModel.java

```

package com.metal.construction.geoIp.model;
import javax.persistence.*;
@Entity
@Table(name = "geoip" , schema = "metal_construction")
public class GeoIpModel {
    @Id
    @Column(name = "geo_ip_id")
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    @Column(name = "country")
    private String country;
    @Column(name = "city")
    private String city;
    @Column(name = "ip")
    private String ip;
    public Long getId() {
        return id;
    }
    public void setId(Long id) {

```

```

        this.id = id;
    }
    public String getCountry() {
        return country;
    }
    public void setCountry(String country) {
        this.country = country;
    }
    public String getCity() {
        return city;
    }
    public void setCity(String city) {
        this.city = city;
    }
    public String getIp() {
        return ip;
    }
    public void setIp(String ip) {
        this.ip = ip;
    }
}

```

Листинг Б.4 – Код файла GeoIpRepository.java

```

package com.metal.construction.geoIp.repository;

import com.metal.construction.geoIp.model.GeoIpModel;
import org.springframework.data.jpa.repository.Modifying;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.CrudRepository;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;

import javax.transaction.Transactional;

@Repository
public interface GeoIpRepository extends
    CrudRepository<GeoIpModel, Long> {
    @Query("Select g from GeoIpModel g where g.ip = :ip")
    GeoIpModel findGeoIpModelByIp(@Param("ip") String ip);

    @Modifying
    @Query(value = "insert into geoip(country, city, ip)
values(:county,:city, :ip)", nativeQuery = true)
    @Transactional
    int insertIp(@Param("ip") String ip, @Param("county") String
country, @Param("city") String city);
}

```

Листинг Б.5 – Код файла GeoIpService.java

```

package com.metal.construction.geoIp.service;

import com.fasterxml.jackson.databind.JsonNode;

```

```

import com.maxmind.db.CHMCache;
import com.maxmind.db.Reader;
import com.metal.construction.geoIp.config.GeoIpConfiguration;
import com.metal.construction.geoIp.model.GeoIpInfo;
import com.metal.construction.geoIp.model.GeoIpModel;
import com.metal.construction.geoIp.repository.GeoIpRepository;
import org.apache.log4j.Logger;
import org.springframework.core.io.ClassPathResource;
import org.springframework.stereotype.Service;
import org.springframework.util.StringUtils;
import javax.annotation.Nullable;
import javax.servlet.http.HttpServletRequest;
import java.io.File;
import java.io.IOException;
import java.net.InetAddress;

@Service
public class GeoIpService implements IGeoIpService{
    private static final Logger logger =
        Logger.getLogger(GeoIpService.class);
    private final Reader reader;

    private final GeoIpRepository geoIpRepository;

    public GeoIpService(GeoIpConfiguration geoIpConfiguration,
        GeoIpRepository geoIpRepository) {
        this.reader = initReader(geoIpConfiguration.getDbPath(),
            geoIpConfiguration.getDbCacheSize());
        this.geoIpRepository = geoIpRepository;
    }

    @Nullable
    @Override
    public GeoIpModel tryGetIpModel(HttpServletRequest
        request) {
        GeoIpInfo geoIpInfo = getGeoIpInfoByRequest(request);
        if (geoIpInfo == null) {
            return null;
        }
        logger.info("Geo data from user " + geoIpInfo);

        GeoIpModel geoIpModel = getModelByIp(geoIpInfo.getIp());
        if (geoIpModel == null) {
            geoIpModel = createModelByIp(geoIpInfo.getIp(),
                geoIpInfo.getCity(), geoIpInfo.getCountry());
        }
        return geoIpModel;
    }

    @Nullable
    public GeoIpInfo getGeoIpInfo(String ip) {
        try {
            InetAddress address = InetAddress.getByName(ip);
            JsonNode response = reader.get(address);
            GeoIpInfo geoIpInfo = createGeoIpInfo(response);
            geoIpInfo.setIp(ip);
        }
    }

```

```

        return geoIpInfo;
    } catch (IOException e) {
        logger.error("error get ip info ip = " + ip, e);
        return null;
    }
}

@Nullable
private GeoIpInfo getGeoIpInfoByRequest(HttpServletRequest request) {
    String ip = getIpFrom(request);
    return ip == null ? null : getGeoIpInfo(ip);
}

@Nullable
private GeoIpModel getModelByIp(String ip) {
    return geoIpRepository.findGeoIpModelByIp(ip);
}

private GeoIpModel createModelByIp(String ip, String city,
String county) {
    GeoIpModel geoIpModel = new GeoIpModel();
    geoIpModel.setIp(ip);
    geoIpModel.setCity(city);
    geoIpModel.setCountry(county);
    Long id = (long) geoIpRepository.insertIp(ip, county,
city);
    geoIpModel.setId(id);
    return geoIpModel;
}

private void getModel(GeoIpModel geoIpModel) {
    geoIpRepository.save(geoIpModel);
}

@Nullable
private String getIpFrom(HttpServletRequest request) {
    String ip = request.getHeader("X-Real-IP");
    if (StringUtils.isEmpty(ip)) {
        ip = request.getRemoteAddr();
    }
    return ip;
}

private GeoIpInfo createGeoIpInfo(JsonNode response) {
    String country = getByPath(response,
"country.iso_code");
    String city = getByPath(response, "city.names.en");
    return new GeoIpInfo(country, city);
}

private String getByPath(JsonNode jsonNode, String path) {
    try {
        String[] parts = path.split("\\.");
        for (int i = 0; i < parts.length; i++) {
            jsonNode = jsonNode.get(parts[i]);
            if (i == parts.length - 1) {
                return jsonNode.asText();
            }
        }
    }

```

```

    }
    } catch (Exception e) {
        logger.trace("element not found for: " + path);
    }
    return "";
}

private Reader initReader(String dbPath, int dbCacheSize) {
    try {
        File file = new ClassPathResource(dbPath).getFile();
        logger.warn(file.getCanonicalPath());
        return new Reader(file, new CHMCache(dbCacheSize));
    } catch (Exception e) {
        throw new RuntimeException("error initialize
database", e);
    }
}
}

```

Листинг Б.6 – Код файла PageController.java

```

package com.metal.construction.page.controller;
import com.metal.construction.geoIp.model.GeoIpModel;
import com.metal.construction.geoIp.service.IGeoIpService;
import com.metal.construction.page.model.PageModel;
import com.metal.construction.page.service.PageService;
import com.metal.construction.visiter.service.VisitorService;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import javax.servlet.http.HttpServletRequest;
@RestController
@RequestMapping("/page")
public class PageController {
    private static final Logger logger =
Logger.getLogger(PageController.class);
    private final PageService pageService;
    private final IGeoIpService geoIpService;
    private final VisitorService visitorService;

    @Autowired
    public PageController(PageService pageService, IGeoIpService
geoIpService, VisitorService visitorService) {
        this.pageService = pageService;
        this.geoIpService = geoIpService;
        this.visitorService = visitorService;
    }
    @PostMapping("/logPage")
    public void logVisitPage(@RequestParam String pageName,
HttpServletRequest request) {

```

```

        GeoIpModel geoIpModel =
geoIpService.tryGetIpModel(request);
        if (geoIpModel == null) {
            logger.info("can't get or create geoip Model");
            return;
        }

        PageModel pageModel =
pageService.getPageByName(pageName);
        if (pageModel == null) {
            pageModel = pageService.createNewPage(pageName);
        }
        visitorService.logVisitor(geoIpModel.getId(),
pageModel.getId(), null);
    }
}

```

Листинг Б.7 – Код файла PageController.java

```

package com.metal.construction.page.controller;
import com.metal.construction.geoIp.model.GeoIpModel;
import com.metal.construction.geoIp.service.IGeoIpService;
import com.metal.construction.page.model.PageModel;
import com.metal.construction.page.service.PageService;
import com.metal.construction.visiter.service.VisitorService;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

import javax.servlet.http.HttpServletRequest;

@RestController
@RequestMapping("/page")
public class PageController {
    private static final Logger logger =
Logger.getLogger(PageController.class);
    private final PageService pageService;
    private final IGeoIpService geoIpService;
    private final VisitorService visitorService;

    @Autowired
    public PageController(PageService pageService, IGeoIpService
geoIpService, VisitorService visitorService) {
        this.pageService = pageService;
        this.geoIpService = geoIpService;
        this.visitorService = visitorService;
    }

    @PostMapping("/logPage")
    public void logVisitPage(@RequestParam String pageName,
HttpServletRequest request) {

```

```

        GeoIpModel geoIpModel =
geoIpService.tryGetIpModel(request);
        if (geoIpModel == null) {
            logger.info("can't get or create geoip Model");
            return;
        }
        PageModel pageModel =
pageService.getPageByName(pageName);
        if (pageModel == null) {
            pageModel = pageService.createNewPage(pageName);
        }
        visitorService.logVisitor(geoIpModel.getId(),
pageModel.getId(), null);
    }
}

```

Листинг Б.8 – Код файла PageModel.java

```

package com.metal.construction.page.model;
import javax.persistence.*;
@Entity
@Table(name = "page", schema = "metal_construction")
public class PageModel {
    @Id @Column(name = "page_id")
    @GeneratedValue(strategy = GenerationType.AUTO)
    private int id;
    @Column(name = "page_name")
    private String pageName;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getPageName() {
        return pageName;
    }
    public void setPageName(String pageName) {
        this.pageName = pageName;
    }
}

```

Листинг Б.9 – Код файла PageRepository.java

```

package com.metal.construction.page.repository;
import com.metal.construction.page.model.PageModel;
import org.springframework.data.jpa.repository.Modifying;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.CrudRepository;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;
import javax.transaction.Transactional;
@Repository

```

```

public interface PageRepository extends
CrudRepository<PageModel, Long> {
    @Query("Select page from PageModel  page where page.pageName
= :name")
    PageModel getPageByName(@Param("name") String name);
    @Modifying
    @Query(value = "INSERT INTO page( page_name)
values(:pageName) ", nativeQuery = true)
    @Transactional
    int insertNewPage(@Param("pageName") String pageName);
}

```

Листинг Б.10 – Код файла PageService.java

```

package com.metal.construction.page.service;
import com.metal.construction.page.model.PageModel;
import com.metal.construction.page.repository.PageRepository;
import org.apache.log4j.Logger;
import org.springframework.stereotype.Service;
import javax.annotation.Nullable;
@Service
public class PageService {
    private static final Logger logger =
Logger.getLogger(PageService.class);

    private final PageRepository pageRepository;
    public PageService(PageRepository pageRepository) {
        this.pageRepository = pageRepository;
    }
    @Nullable
    public PageModel getPageByName(String pageName) {
        return pageRepository.getPageByName(pageName);
    }
    public PageModel createNewPage(String pageName) {
        PageModel pageModel = new PageModel();
        pageModel.setPageName(pageName);
        logger.info("Created page model" + pageModel);

        pageModel.setId(pageRepository.insertNewPage(pageModel.getPageName()));

        return pageModel;
    }
}

```

Листинг Б.10 – Код файла ProductService.java

```

package com.metal.construction.product.controller;

import com.google.common.collect.Streams;
import com.metal.construction.product.dto.ProductDto;
import com.metal.construction.product.service.ProductService;
import org.springframework.beans.factory.annotation.Autowired;

```

```

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import java.util.List;
import java.util.stream.Collectors;
@RestController
@RequestMapping("/product")
public class ProductController {

    private final ProductService productService;

    @Autowired
    public ProductController(ProductService productService) {
        this.productService = productService;
    }

    @GetMapping("/getAllProducts")
    public List<ProductDto> getProducts() {
        return
Streams.stream(productService.getAvailableProduct()).
        map(productModel ->
ProductDto.create(productModel))
        .collect(Collectors.toList());
    }
}

```

Листинг Б.11 – Код файла LoginInPage.jsx

```

import React, {useEffect} from 'react';
import Avatar from '@material-ui/core/Avatar';
import Button from '@material-ui/core/Button';
import CssBaseline from '@material-ui/core/CssBaseline';
import TextField from '@material-ui/core/TextField';
import FormControlLabel from '@material-
ui/core/FormControlLabel';
import Checkbox from '@material-ui/core/Checkbox';
import Grid from '@material-ui/core/Grid';
import Box from '@material-ui/core/Box';
import LockOutlinedIcon from '@material-ui/icons/LockOutlined';
import Typography from '@material-ui/core/Typography';
import { makeStyles } from '@material-ui/core/styles';
import Container from '@material-ui/core/Container';
import Copyright from "../component/copyright/Copyright";
import {Link} from "react-router-dom";
import LogPage from "../api/PageLogApi";
import axios from "../core/AxiosApi";

export default function LoginIn() {
    const classes = useStyles();
    useEffect(() => {
        LogPage("LoginIn");
    });

    return (

```

```

<Container component="main" maxWidth="xs">
  <CssBaseline />

  <div className={classes.paper}>
    <Avatar className={classes.avatar}>
      <LockOutlinedIcon />
    </Avatar>
    <Typography component="h1" variant="h5">
      Войти
    </Typography>
    <form className={classes.form} noValidate>
      <TextField
        variant="outlined"
        margin="normal"
        required
        fullWidth
        id="email"
        label="Email Address"
        name="email"
        autoComplete="email"
        autoFocus
      />
      <TextField
        variant="outlined"
        margin="normal"
        required
        fullWidth
        name="password"
        label="Пароль"
        type="password"
        id="password"
        autoComplete="current-password"
      />
      <FormControlLabel
        control={<Checkbox value="remember"
color="primary" />}
        label="Запомнить меня"
      />
      <Button
        type="submit"
        fullWidth
        variant="contained"
        color="primary"
        className={classes.submit}
      >
        Войти
      </Button>
      <Grid container>
        <Grid item xs>
          <Link to = "/recoverAccount"
variant="body2">
            Забыли пароль?
          </Link>
        </Grid>

```

```

                                <Grid item xs>
                                    <Link to = "/registration"
variant="body2">
                                Еще не создали аккаунт?
                                    {"\n"}Зарегистрируйтесь
                                </Link>
                            </Grid>
                        </Grid>
                    </form>
                </div>
                <Box mt={8}>
                    <Copyright />
                </Box>
            </Container>
        );
    }

```

Листинг Б.12 – Код файла MainPage.jsx

```

import '../App.css';
import Album from "../component/album/Album";

import {useEffect} from "react";
import LogPage from "../api/PageLogApi";
import LogReference from "../api/ReferenceUserApi";

function MainPage() {

    useEffect(() => {
        console.log(document.referrer)
        if(document.referrer === null || document.referrer ===
"" ) {
            LogPage("MainPage");
        }else{
            LogReference(document.referrer, "MainPage")
        }
    });
    return (
        <div className="MainPage">
            <Album/>
        </div>
    );
}
export default MainPage;

```

Листинг Б.11 – Код файла AlbumPage.jsx

```

import React, {Component} from 'react'
import AppBar from '@material-ui/core/AppBar';
import Button from '@material-ui/core/Button';
import Card from '@material-ui/core/Card';
import CardActions from '@material-ui/core/CardActions';
import CardContent from '@material-ui/core/CardContent';

```

```

import CardMedia from '@material-ui/core/CardMedia';
import CssBaseline from '@material-ui/core/CssBaseline';
import Grid from '@material-ui/core/Grid';
import Typography from '@material-ui/core/Typography';
import Container from '@material-ui/core/Container';
import Copyright from "../copyright/Copyright";
import {withStyles} from '@material-ui/core/styles';
import albumElements from '../.../testAlbumJson';//тестовая
версия json-данных
import MainPageToolBar from
'../mainPageToolBar/MainPageToolBar';
import axios from "../.../core/AxiosApi";

class Album extends Component {
  constructor(props) {
    super(props);
    this.state = {
      products : []
    };
  }
  componentDidMount() {
    this.getProducts()
      .then( receivedProducts => {
        console.log(receivedProducts);
        this.setState({
          products : receivedProducts
        });
      })
      .catch();

  }
  async getProducts() {
    try{
      const response = await
axios.get("/product/getAllProducts");
      return response.data;
    }catch (error){
      console.error(error);
    }
  }
  render() {
    const products = this.state.products;
    const {classes} = this.props;
    return (
      <React.Fragment>
        <CssBaseline/>
        <MainPageToolBar/>
        <main>
          <div className={classes.heroContent}>
            <Container maxWidth="sm">
              <Typography component="h1"
variant="h2" align="center" color="textPrimary" gutterBottom>
                Stalkon
              </Typography>

```

```

        <Typography variant="h5"
align="center" color="textSecondary" paragraph>
        Откройте для себя восхитительный
мир украшений из серебра и золота!
        Мы заботимся о каждом моменте
вашей жизни, уделяя внимание деталям.
        В дизайне украшений компании
легко найти что-то на свой вкус!
    </Typography>
    <div
className={classes.heroButtons}>
        <Grid container spacing={2}
justify="center">
            <Grid item>
                <Button
variant="contained" color="primary">
                    Связаться с нами
                </Button>
            </Grid>
            <Grid item>
                <Button
variant="outlined" color="primary">
                    Больше товаров в
каталоге
                </Button>
            </Grid>
        </Grid>
    </div>
</Container>
</div>
<Container className={classes.cardGrid}
maxWidth="md">
    { /* End hero unit */ }
    <Grid container spacing={4}>
        {products.map((products) => (
            <Grid item key={products.id}
xs={12} sm={6} md={4}>
                <Card
className={classes.card}>
                    <CardMedia
className={classes.cardMedia}
image=
{products.imagePath}
title=
{products.header}
/>
                    <CardContent
className={classes.cardContent}>
                        <Typography
gutterBottom variant="h5" component="h2">
                            {products.header}
                        </Typography>

```

```

                                <Typography>
{products.descriptionPreview}
                                </Typography>
                                </CardContent>
                                <CardActions>
                                <Grid container
justify="flex-end">
                                <Button
size="small" color="primary" align = "left">
                                Просмотр
                                </Button>
                                </Grid>
                                </CardActions>
                                </Card>
                                </Grid>
                                )})
                                </Grid>
                                </Container>
</main>
{/* Footer */}
<footer className={classes.footer}>
    <Typography variant="h6" align="center"
gutterBottom>
        Stalkon
    </Typography>
    <Typography variant="subtitle1"
align="center" color="textSecondary" component="p">
        Все права защищены
    </Typography>
    <Copyright/>
</footer>
    {/* End footer */}
</React.Fragment>
    );
}
}

export default withStyles(useStyles) (Album)

```

Приложение В
ТЕСТИРОВАНИЕ ПРОГРАММЫ
(рекомендуемое)

Регистрация

Имя *

Суворов

Фамилия *

Максим

Email Address *

royming06@gmail.com

Пароль *

☒ Я хочу получать рассылку об акциях и предложениях

ЗАРЕГИСТРИРОВАТЬСЯ

[Уже есть аккаунт? Войдите](#)

Copyright © Stalkon 2021.

Рисунок В.1 – Форма регистрации

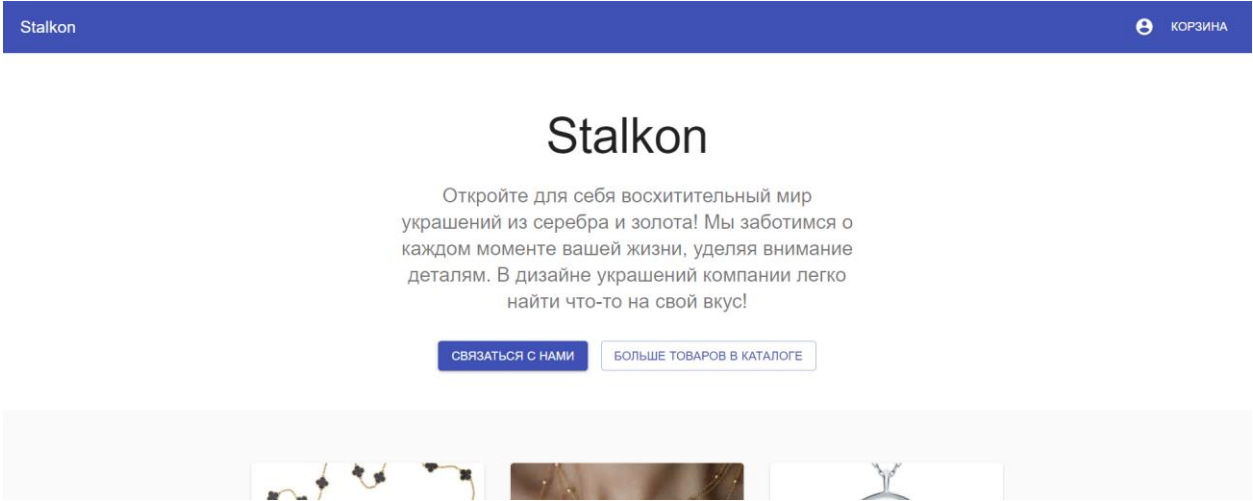


Рисунок В.2 – Главная страница после выполнения регистрации

Войти

Email Address *

royming06@gmail.com

Пароль *

☒ Запомнить меня

ВОЙТИ

[Забыли пароль?](#)

[Еще не создали аккаунт? Зарегистрируйтесь](#)

Copyright © Stalkon 2021.

Рисунок В.3 – Форма аутентификации

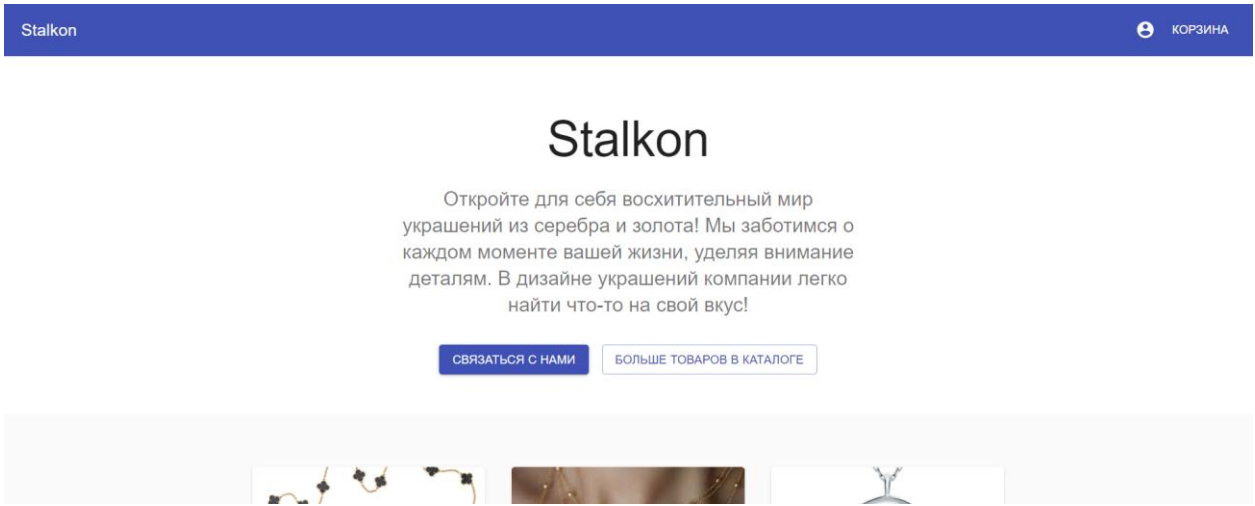


Рисунок В.4 – Главная страница после выполнения регистрации

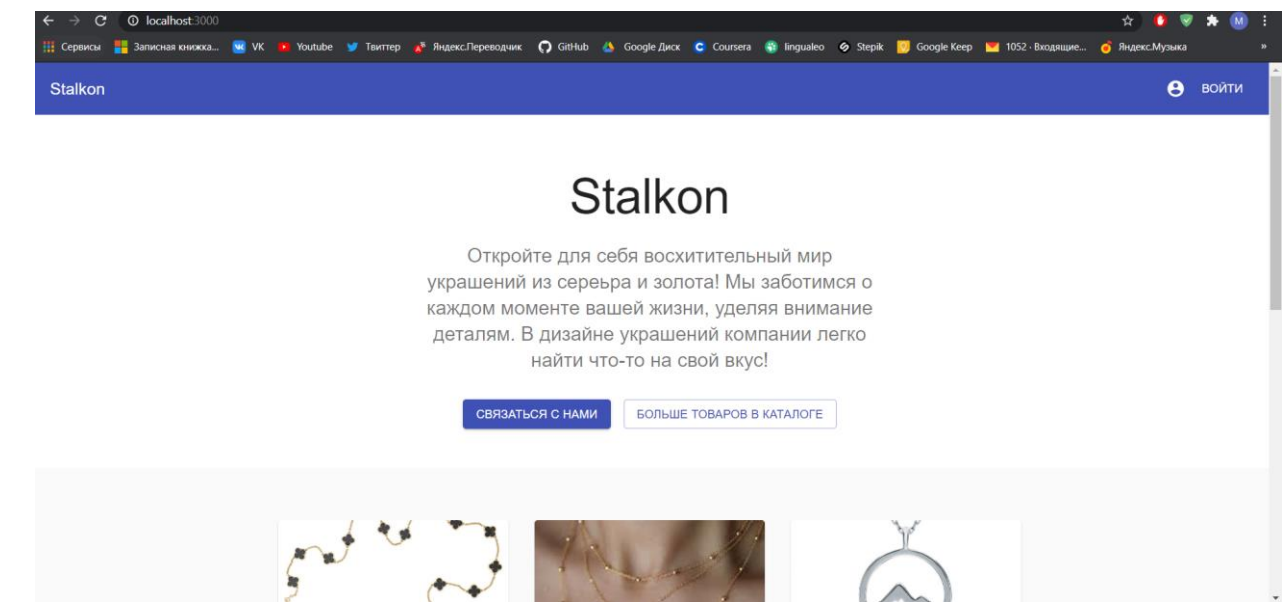


Рисунок В.5 – Главная страница приложения

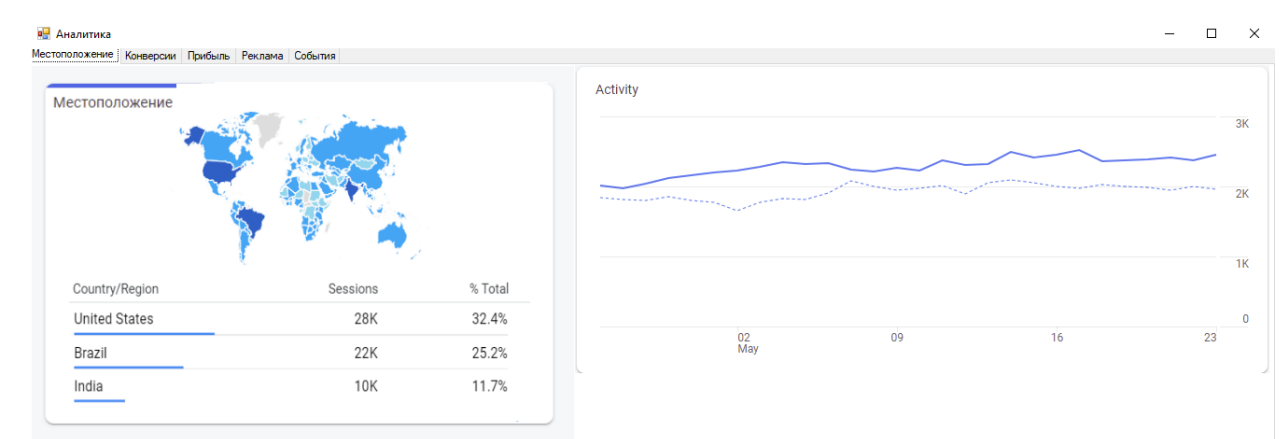


Рисунок В.6 – Страница с анализом географической распределения клиентов

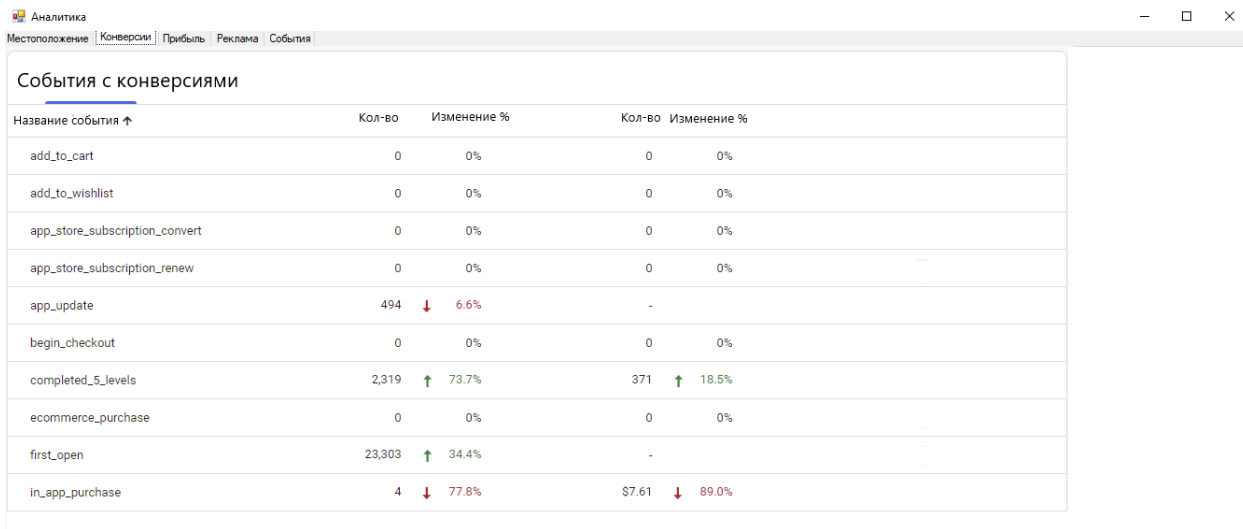


Рисунок В.7 – Анализ конверсий страниц сайта

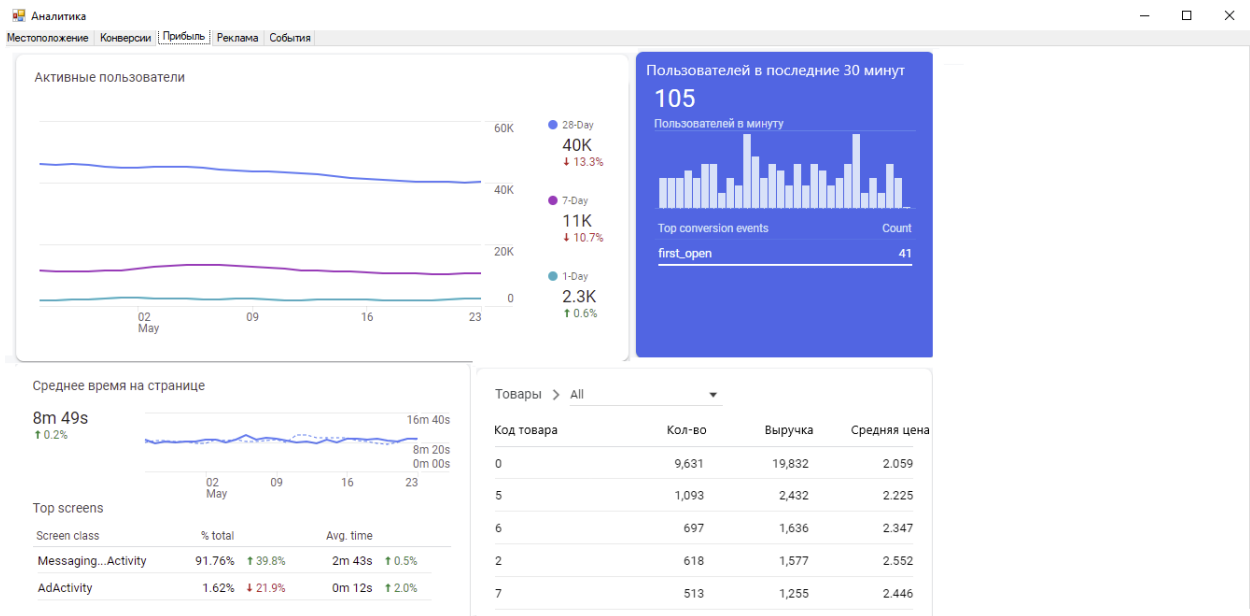


Рисунок В.8 – Анализ прибыли