

СОДЕРЖАНИЕ

1	Определения и сокращения	7
	Введение	8
2	Аналитический обзор существующих решений	9
2.1	Google Analytics	9
2.2	CRM системы	10
2.3	FullContact	17
2.4	Постановка задачи	17
3	Модели, положенные в основу разрабатываемого программного средства	19
3.1	Функциональная модель программного средства	19
3.2	Спецификация функциональных требований	20
3.3	Спецификация нефункциональных требований	21
4	Проектирование программного средства	22
4.1	Архитектура микросервисов	22
4.2	Архитектура программного средства	23
5	Разработка программного средства	25
5.1	Spring Cloud и Netflix OSS	25
5.2	Алгоритм сохранения события	28
5.3	Алгоритм классификации событий	30
5.4	Схема интеграции с онлайн чатом Salesforce	31
6	Тестирование приложения	34
6.1	Тестирование административной панели	34
6.2	Тестирование интеграции с Salesforce CRM	36
7	Руководство пользователя	38
7.1	Руководство по разворачиванию приложения	38
7.2	Руководство для администраторов и менеджеров	39
7.3	Руководство по интеграции с онлайн чатом	40
8	Технико-экономическое обоснование разработки ПС	43
8.1	Расчёт сметы затрат и цены программного продукта	43
8.2	Расчёт трудоемкости	45
8.3	Расчёт заработной платы исполнителей	47
8.4	Расчёт расходов и прогнозируемой цены ПО	49
8.5	Расчёт экономической эффективности у разработчика	50
8.6	Выводы по технико-экономическому обоснованию	51

Заключение	53
----------------------	----

1 ОПРЕДЕЛЕНИЯ И СОКРАЩЕНИЯ

CRM (сокращение от англ. Customer Relationship Management) - прикладное программное обеспечение для организаций, предназначенное для автоматизации стратегий взаимодействия с заказчиками (клиентами), в частности, для повышения уровня продаж, оптимизации маркетинга и улучшения обслуживания клиентов путём сохранения информации о клиентах и истории взаимоотношений с ними, установления и улучшения бизнес-процессов и последующего анализа результатов. Time to live (TTL) — предельный период времени или число итераций или переходов, за который набор данных (пакет) может существовать до своего исчезновения.

SF - сокращенно Salesforce.

Сниппет — это фрагмент исходного кода или текста, пригодный для повторного использования. Сниппеты не являются заменой процедур, функций или других подобных понятий структурного программирования. Они обычно используются для более лёгкой читаемости кода функций, которые без их использования выглядят слишком перегруженными деталями, или для устранения повторения одного и того же общего участка кода.

UI — (англ. user interface) — разновидность интерфейсов, в котором одна сторона представлена человеком (пользователем), другая — машиной/устройством. Представляет собой совокупность средств и методов, при помощи которых пользователь взаимодействует с различными, чаще всего сложными, машинами, устройствами и аппаратурой

ВВЕДЕНИЕ

В связи с развитием информационных технологий в настоящее время для почти любого бизнеса легче и прибыльней всего привлекать клиентов в интернете. Поэтому для успешного ведения бизнеса у любой компании должен быть сайт, предоставляющий всю необходимую информацию потенциальным клиентам. Компании всё больше и больше уделяют внимание построению и улучшению своего сайта, который должен быть легко доступным и интуитивно понятным. В нынешнее время, даже если сайт компании сделан на высоком уровне, с течением времени он устаривает, потому что запросы клиентов растут, в то время как конкуренты не стоят на месте. Поэтому, чтобы не потерять текущих клиентов и привлекать больше новых, нужно постоянно усовершенствовать свой сайт. Анализ поведения клиентов на сайте позволяет лучше определить сферы для улучшения сайта.

Для взаимодействия сайта с клиентом уже существует целый ряд готовых программных средств. Например, различные CRM системы, Google Analytics, сервисы по предоставлению информации о пользователе и многое другое. У всех есть свои особенности и уникальные возможности, поэтому зачастую используются сразу несколько специализированных средств.

К сожалению, существующие системы не всегда легко связать между собой, чтобы получить все их преимущества. У каждой из них своя доменная модель, которая ничего не знает о других системах. Из-за этого каждый раз приходится разрабатывать соответствующую логику в приложении, для их взаимодействия друг с другом.

Целью данного дипломного проекта является создание программного средства,ключающего функционал взаимодействия клиента с сайтом, интегрирования его с различными CRM системами и другими программными средствами для собирания статистики действий пользователя на сайте. Кроме того, должно быть возможно добавление новых событий в любое время администраторами и менеджерами сайта, по которым должен происходить сбор информации. Система должна быть легко расширяема для нового функционала и интеграций. Быть высоко производительной и иметь как можно меньший отклик на события, а так же быть отказоустойчивой. Приспособлена к работе с большим количеством поступающих входных данных и иметь интуитивно понятный интерфейс.

2 АНАЛИТИЧЕСКИЙ ОБЗОР СУЩЕСТВУЮЩИХ РЕШЕНИЙ

Существует целый ряд готовых средств для того, чтобы взаимодействовать с клиентом на сайте и получать о нем как можно более точный профиль данных, например различные CRM системы, Google Analytics и многое другое. У всех есть свои особенности и уникальные возможности, поэтому зачастую используются сразу несколько специализированных средств.

2.1 Google Analytics

Самым популярным средством для сбора статистики данных о пользователе в данный момент является Google Analytics(рисунок 2.1), которая включает в себя обширные возможности.

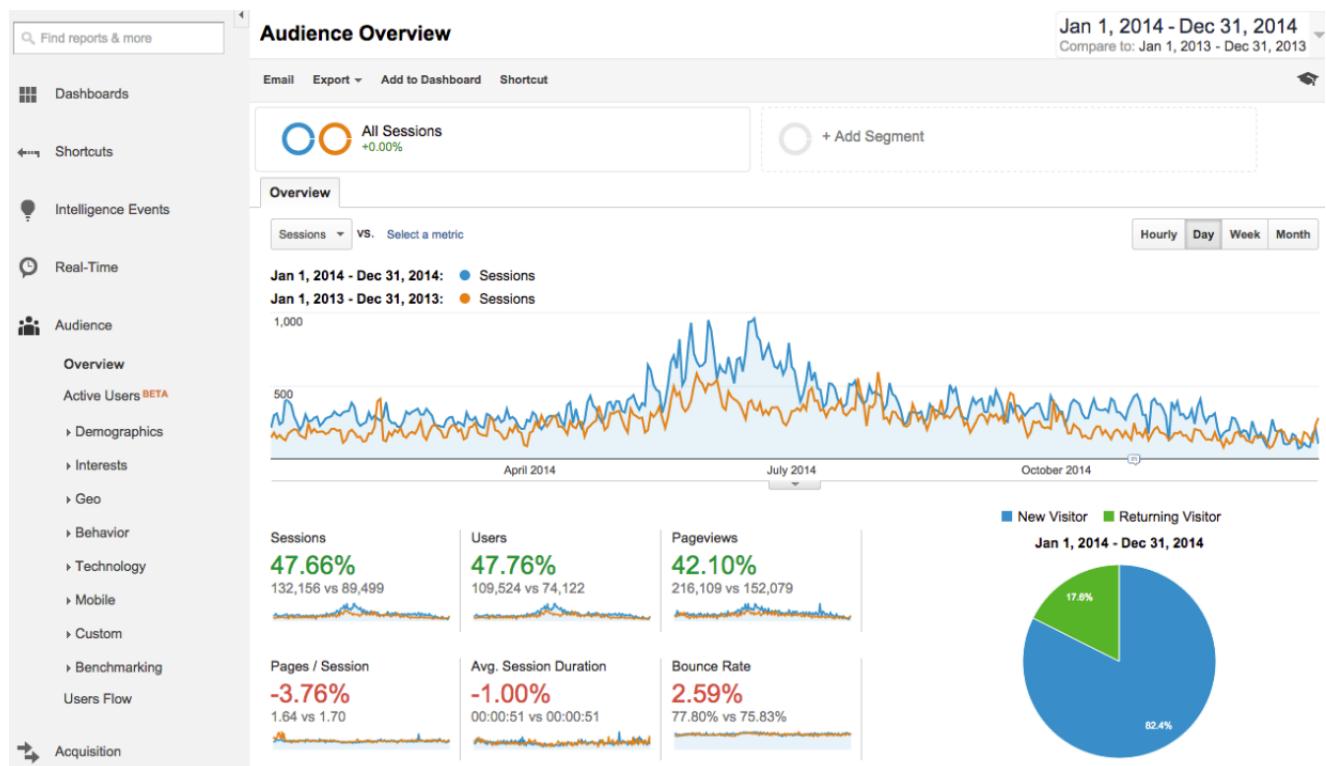


Рисунок 2.1 – Google Analytics

Преимущества:

- способность отслеживать статистику переходов на сайт;
- способность классифицировать посетителей, что позволяет разрабатывать новые страницы сайта более эффективно с учетом потребностей целевой аудитории;

- способность отслеживать исходящие ссылки, используемые при продвижении сайта, также необходимые для дальнейшей раскрутки сайта;
- способность отслеживать ссылки, которые скачивают больше всего;
- способность отслеживать адреса электронной почты по кликам;
- практически все коммерческие транзакции прослеживаются при помощи Google Analytics;
- можно отключить статистику посещений тех, кто обслуживает сайт;
- Сравнение статистики посещения сайта за разные периоды. Данная возможность идеальна для выявления эффективности работы новых страниц.

Недостатки:

- неизвестно отследить трафик если у пользователя отключены cookies;
- Google Analytics не может повторно обработать данные, если потерян профиль с настроенными фильтрами;
- настройка отчетов имеет ограниченное количество;
- количество отслеживаемых целей также ограничено (настоящее время Google Analytics отслеживает до четырех целей).

2.2 CRM системы

Раньше CRM системы были доступны только для корпоративного сектора. Исключительно компании с развитой технологической инфраструктурой, огромным штатом сотрудников и достаточным бюджетом могли приобрести CRM систему для автоматизации работы отделов продаж, маркетинга и сервисного обслуживания клиентов. Постепенно с тем, как увеличивалась скорость Интернет-подключения, появлялись облачные технологии и приобретали свою популярность SaaS решения (англ. software as a service — программное обеспечение как услуга), приобретение CRM систем становилось доступной опцией для любой компании.

Сегодня рынок CRM является динамичным и быстрорастущим. Облачные технологии позволяют легко внедрять CRM системы с нуля, без особых технических хлопот с развертыванием и по низкой стоимости. Далее приведены основные характеристики CRM систем, покупательские критерии, а также сравнения CRM решений от разных поставщиков. Конечно есть свои нюансы, и бизнесы должны полагаться на свои собственные требования при выборе оптимальной CRM системы. Далее рассмотрены базовые параметры, на которые следует обратить внимание при выборе CRM системы:

- варианты хостинга;
- наличие мобильного клиента;

- стоимость;
- функционал для продаж;
- функционал для маркетинга;
- наличие сервисного модуля;
- возможности хранения документов на дисковых пространствах CRM системы;
- модуль отчетности.

Далее приведен обзор следующих CRM систем: SugarCRM, Salesforce.com, Microsoft Dynamics CRM и Zoho.

2.2.1 SugarCRM

SugarCRM (рисунок 2.2) предлагает систему для поддержки процессов продаж, маркетинга и сервисного обслуживания. Стоимость лицензии начинается от \$35 за пользователя в месяц в рамках редакции Professional с возможностью развертывания решения на серверах предприятия и до \$150 за пользователя в месяц за редакцию Ultimate. В рамках подписки на все редакции предоставляется мобильный клиент, а размеры хранилища документов в «облаке» варьируются от 15 Гб в редакции Professional до 250 ГБ – в Ultimate.

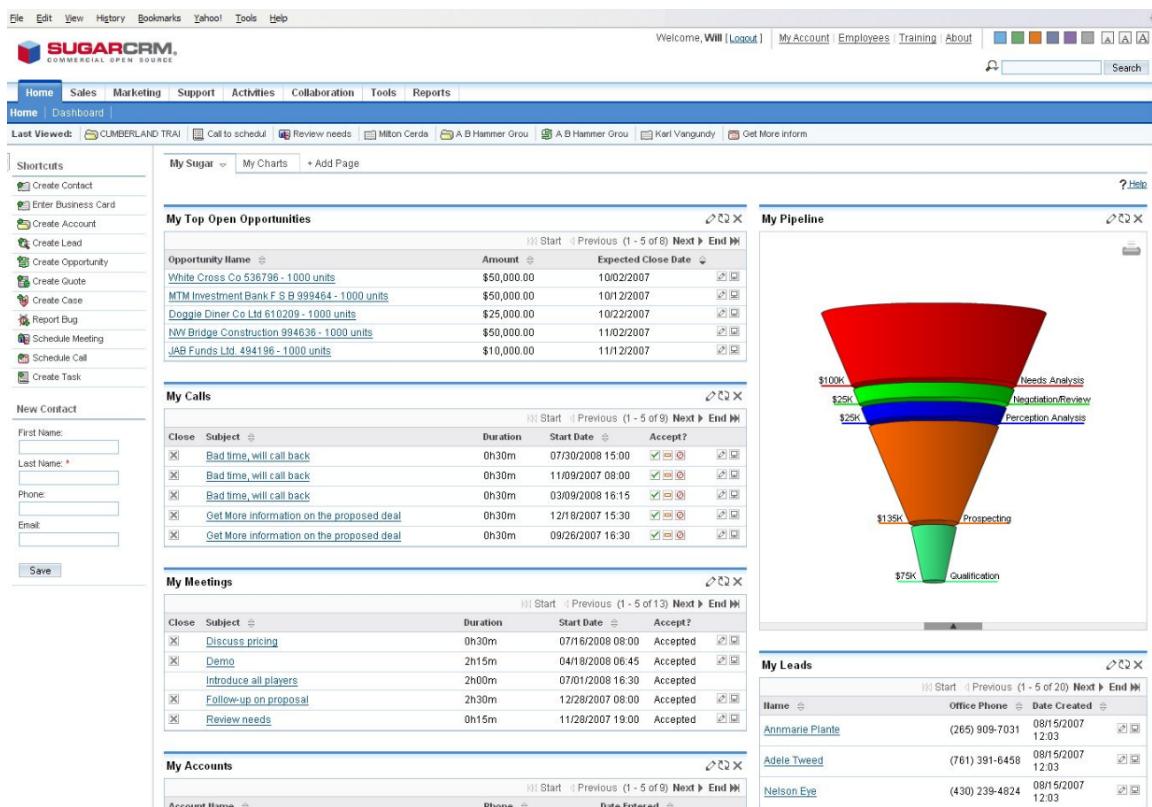


Рисунок 2.2 – SugarCRM

Основные функциональные возможности SugarCRM:

– Workflow или автоматические процессы. Позволяет автоматически настраивать события и следить за их выполнением. В продажах этот механизм особенно важен: позволяет создавать новые продажи с различным бюджетом, контролировать пребывание продажи на конкретной стадии и многое другое.

– Интеграция с SMS. Как и интеграция с социальными сетями стала неотъемлемой частью работы компании, SMS сервисы стали важными каналами коммуникации с клиентами и предоставления сервиса последним. Возможность интегрировать CRM систему и сервис SMS сообщений является важным преимуществом.

– Функция Web-to-lead. CRM система Sugar также позволяет собирать клиентские данные с веб-сайта компании и автоматически аккумулировать эти данные в CRM системе.

Преимущества:

– SugarCRM имеет открытый исходный код, поэтому система достаточно гибкая и масштабируемая с возможностью расширения под потребности бизнеса;

– систему можно достаточно легко настраивать под предпочтения бизнеса пользователей.

Недостатки:

– чтобы пользоваться SugarCRM нужны определенные знания и компетенции, на обучение требуется время.

2.2.2 Salesforce CRM

Salesforce CRM (рисунок 2.3) лидирует в сегменте облачных CRM систем уже на протяжении многих лет. Salesforce.com предлагает CRM систему Sales Cloud, которая поставляется в пяти редакциях, начиная с базовой редакции Contact Manager стоимость лицензии которой начинается от \$5 за пользователя в месяц. Наиболее популярная редакция, которая наилучшим образом удовлетворяет бизнес потребности большинства компаний, это Enterprise. Ее стоимость составляет \$125 за пользователя в месяц; редакция поставляется с функционалом по ведению и управлению продажами, управлению процессами lead менеджмента, настраиваемыми рабочими столами (dashboard), workflow и возможностями интеграции через API.

Основные функциональные возможности Salesforce.com:

– Синхронизация с Outlook. Данные из CRM системы Salesforce автоматически синхронизируются с Outlook, включая контакты, календарь и многое другое.

– Настраиваемые процессы продаж. Можно адаптировать процессы про-

даж под модель организации компании. Это играет ключевую роль при выборе CRM системы, потому что продавцам очень важно использовать технологию, в которой доступны не только стандартные процессы.

- Функция Web-to-lead. Эта функция позволяет компаниям собирать информацию со своих сайтов и генерировать ее внутри CRM системы Salesforce, на основе этих данных создавать лиды.
- Мобильный доступ в режиме offline. Очень важная опция для полевых продавцов, которые могут вводить данные в CRM систему с мобильного в автономном режиме и позже синхронизировать эти данные в CRM при возобновлении подключения к Интернет.
- Интеграция через web-сервисы API. Эта опция позволяет CRM системе Salesforce синхронизироваться с другими backend офисными системами, такими как ERP, системами финансового учета, а также дает возможность компаниям расширять функциональность и интегрировать систему с другими технологиями.

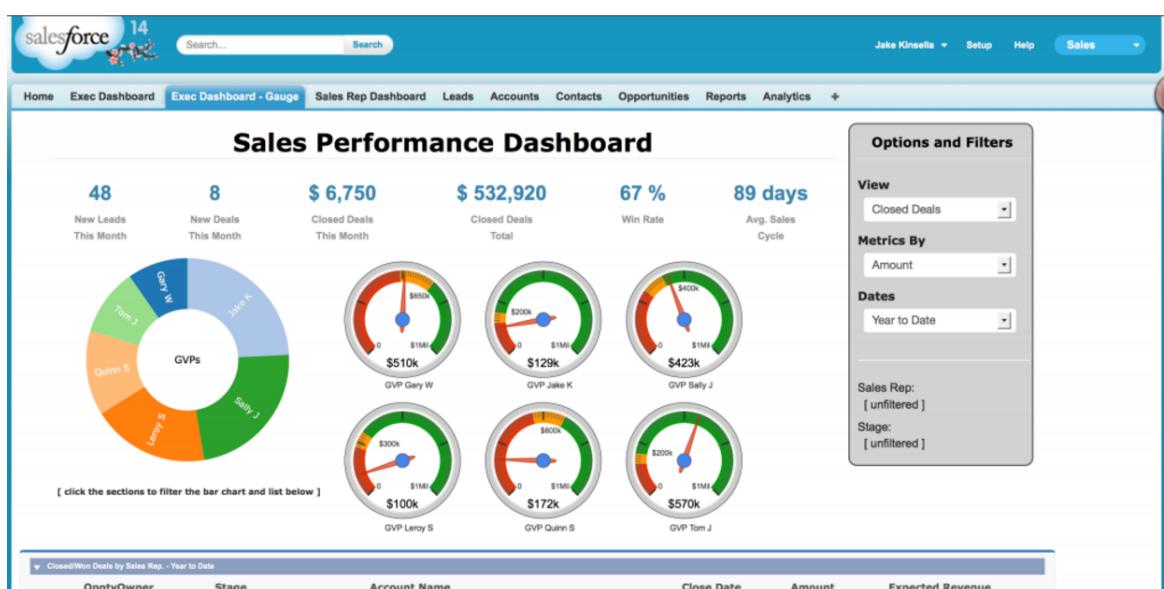


Рисунок 2.3 – Salesforce CRM

Преимущества:

- гибкость системы, наличие ключевых CRM функций, в том числе визуализация воронки продаж в режиме реального времени;
- мобильный клиент для всех редакций системы;
- возможная интеграция с инструментами работы с данными от третьих производителей, такими как Data.com, Twitter, LinkedIn, YouTube и Klout, увеличит производительность работы на всех этапах цикла продажи, от потенциального клиента до клиента.

Недостатки:

- только облачный вариант развертывания, что ставит под вопрос безопасность клиентских данных для некоторых компаний;
- стоимость редакций Professional и Enterprise дороже, чем у большинства других игроков рынка.

2.2.3 Microsoft Dynamics CRM

Microsoft Dynamics CRM (рисунок 2.4) поздно появилась на рынке CRM, и ранние редакции Microsoft CRM не имели большого спроса среди пользователей. У Microsoft довольно сложная процедура лицензирования, поэтому цены данного обзора основаны на облачном предложении от Microsoft - CRM online. Стоимость этой редакции начинается от \$65 за пользователя в месяц. Dynamics CRM – это технология с полным набором различных функций - от управления лидами и до заключения продажи, поэтому с помощью Dynamics CRM можно выстраивать полноценные отношения с клиентами. CRM система интегрируется с другими инструментами от Microsoft – программным пакетом Office и Office 365 – для ведения email коммуникаций, анализа данных и управления документами – однако это все за дополнительную плату.

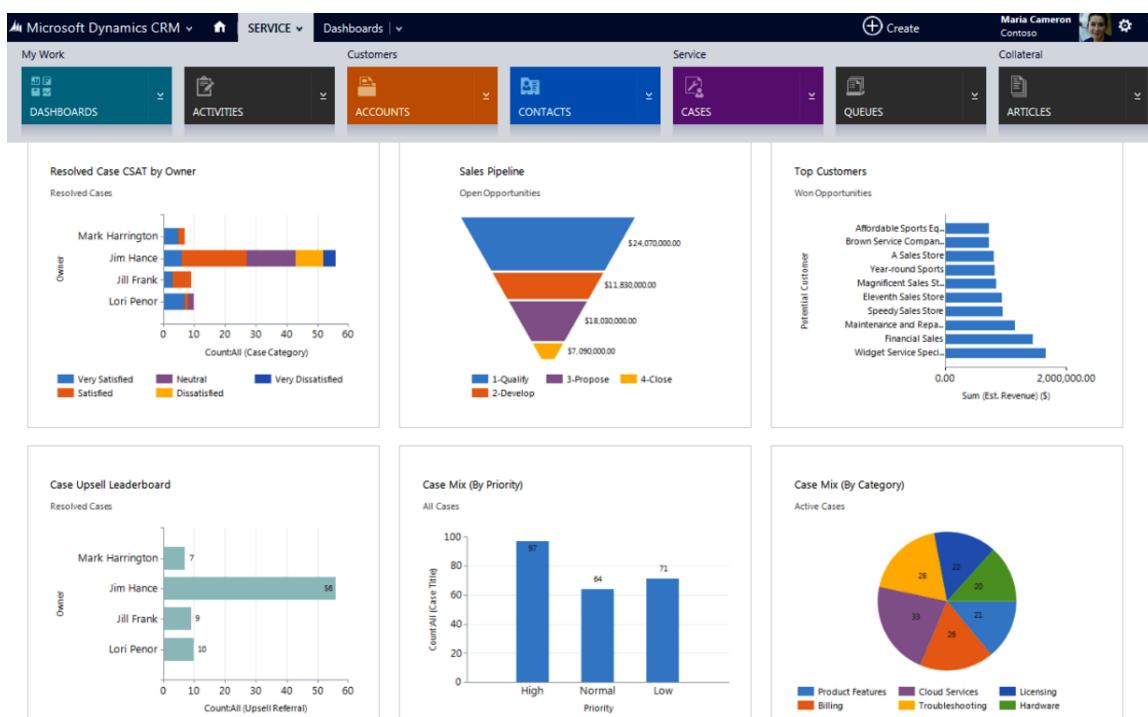


Рисунок 2.4 – Microsoft Dynamics CRM

Основные функциональные возможности Dynamics CRM:

- интеграция с Microsoft Office;

- Настраиваемые отчеты и рабочие столы. Есть возможность генерировать кастомизированные отчеты для руководства.
- Механизм workflow для настройки бизнес процессов. Можно планировать и оптимизировать бизнес-процессы в CRM системе Dynamics с помощью данного инструмента.
- Интеграция через web-сервисы. Microsoft синхронизируется с другими бэк оффисными системами, такими как ERP системы, системы финансового учета и др.
- Мобильный доступ. Microsoft разработала систему Dynamics с доступной мобильной версией.
- Кастомизированные сущности. Эта функция позволяет третьим производителям дорабатывать функционал готовой CRM системы. Например, кастомизированная сущность "Проект" может быть создана для управления отношениями между контактами и контрагентами с поддержкой функционала существующих сущностей системы.
- Соглашение о качестве предоставляемых услуг. Microsoft подписывает соглашение о качестве предоставляемых услуг, которое обеспечивает компаниям безопасность клиентских данных, а также страхует их от потери данных и других потенциальных угроз.

Преимущества:

- Microsoft обладает богатым функционалом для увеличения лидов до продажи или сервисного обращения;
- Microsoft хорошо интегрируется с другими продуктами для повышения производительности бизнеса, такими как Office и Office 365.

Недостатки:

- решение Microsoft CRM стоит очень дорого;
- известно больше как преемник тенденций, а не новатор.

2.2.4 Zoho CRM

Zoho CRM (рисунок 2.5) предлагает различные онлайн продукты и облачные технологии; CRM система – одна из них. Стоимость владения системой довольно низкая; система поставляется на бесплатной основе, если количество пользователей не превышает 3 единицы, и может послужить хорошей отправной точкой для представителей малого бизнеса. На таких условиях предоставляется базовый функционал управления лидами, продажами, контрагентами и контактами.

The screenshot displays the Zoho Social dashboard. On the left, a sidebar includes links for Home, Posts, Messages, Monitor, Connections, Collaborate, and Reports. The main area features two sections: 'Brand Health' and 'Brand Index'. 'Brand Health' provides a table of audience metrics for various networks (Facebook, Twitter, LinkedIn, YouTube) across different time periods. 'Brand Index' is a bar chart showing engagement levels over time. To the right, a 'Live Stream' section shows a feed of social media posts from various users.

Networks	Total Audience ?	Active Audience ?	Engagement ?	Stories Created ?
Zylker	31,293 +0.05%	3,193 -1.65%	17,858 -0.6%	1,561 +1.61%
Zylker	17,807 +0.53%	1,743 +2.9%	493 +4.78%	241 +5.13%
Zylker Inc.	17,931 +1.66%	637 +53.85%	163 +0%	96 +0%
Zylker	525 +9.8%	35 +3.3%	84 +9.6%	43 +41.18%

Рисунок 2.5 – Zoho CRM

Основные функциональные возможности Zoho:

- Функция Web to lead, case формы. Пользователи могут собирать данные из форм непосредственно в CRM систему Zoho.
- Настраиваемые рабочие столы. На рабочий стол пользователи могут выводить нужную им информацию для оперативной работы.
- Автоматизация маркетинга. Инструмент автоматизации маркетинга автоматически сегментирует целевую аудиторию компании для точечного обращения и позволяет измерять затраченные усилия.
- Интеграция с Twitter и Facebook. Увеличение роли социальных сетей в улучшении клиентского опыта. Необходимость отслеживать поведения клиентов в социальных медиа, а также потребность в повышении эффективности маркетинга, делают интеграцию с социальными платформами обязательным элементом для CRM систем.

Преимущества:

- сбор данные веб-форм сайта непосредственно в CRM систему;
- можно попробовать CRM систему перед тем, как приобретать ее;
- Zoho является достойным продуктом и по стоимости уступает большинству.

Недостатки:

- бесплатная редакция хороша в качестве пробы, но у нее есть жесткое ограничение по количеству данных, которые могут храниться в системе.

2.3 FullContact

FullContact (рисунок 2.6) позволяет легко искать информацию о пользователе по email, телефонном номере или по имени аккаунта в твиттере. Он позволяет найти всю публичную информацию из доступных социальных сетей, фотографий, географического положения, карьере и около 100 других различных данных о пользователе.

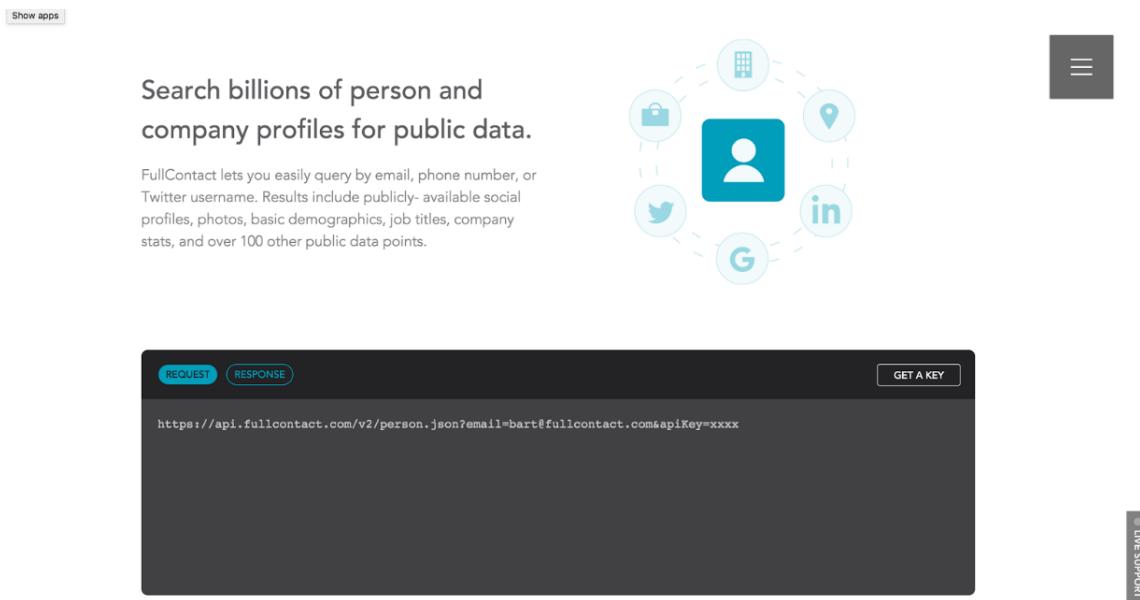


Рисунок 2.6 – FullContact

Преимущества:

- предоставляет всевозможную публичную информацию об интересующих объектах;
- простая интеграция.

Недостатки:

- является всего лишь сервисом по предоставлению публичной информации;
- оплата зависит от количества произведенных запросов.

2.4 Постановка задачи

Таким образом, проанализировав существующие готовые решения, было решено разработать программное средство предоставляющие возможность объединить их преимущества вместе и синтегрировать их все в одну платформу. Оно должно обладать следующими свойствами:

- легко интегрироваться с другими программными средствами и горизонтально масштабироваться;
- иметь интеграцию с онлайн чатом CRM Salesforce;
- иметь функционал трекинга событий произведенных пользователем на сайте;
- иметь панель администратора, включающую в себя: просмотр статистики по пользователям как для выбранного периода времени, так и в режиме онлайн, возможность добавления администраторами и менеджерами новых событий в любое время;
- интеграция с FullContacts для собирания публичной информации о пользователе из доступных социальных сетей, фотографий, географического положения, карьере и других различных данных о пользователе;
- отказоустойчивость.

3 МОДЕЛИ, ПОЛОЖЕННЫЕ В ОСНОВУ РАЗРАБАТЫВАЮЩЕГО ПРОГРАММНОГО СРЕДСТВА

3.1 Функциональная модель программного средства

Для представления функциональной модели была выбрана диаграмма вариантов использования UML, которая отражает отношения между актерами и прецедентами и позволяет описать систему на концептуальном уровне.

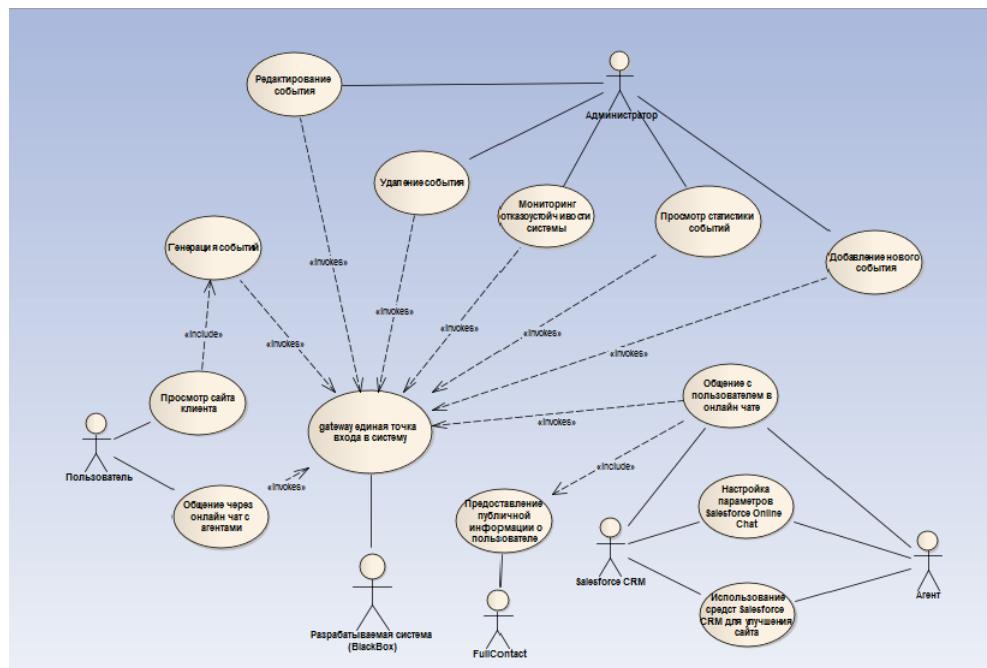


Рисунок 3.1 – Функциональная модель

Функциональная модель представлена на (рисунке 3.1).

На основании представленной диаграммы функциональной модели можно сделать вывод, что в системе будет существовать как минимум 6 актеров:

- пользователь;
- агент;
- администратор;
- Salesforce CRM;
- FullContact;
- разрабатываемая система (BlackBox).

Рассмотрим каждого актера с его функциями более подробно.

Пользователь - у пользователя в данной системе только две функции: он просматривает сайт (и тем самым генерирует события) и общается в онлайн чате с агентом.

Агент - агент работает в Salesforce CRM. Он настраивает параметры для Salesforce Online Chat для отдельных страниц сайта, со своими особыми правилами появления чата, непосредственно общается с пользователем сайта, чтобы помочь последним и использует средства Salesforce CRM для улучшения сайта в результате диалога с пользователем.

Администратор - настраивает систему отслеживания событий (добавляет/удаляет/редактирует события), следит и просматривает статистики происходящих событий для того чтобы понять как можно улучшить сайт. Также администратор мониторит отказоустойчивость системы, чтобы в случае непредвиденных сбоев (потеря сети или интернета, выхода из строя серверов) быть готовым среагировать на возникшую проблему.

Salesforce CRM - внешняя система, благодаря интеграции с которой, возникает возможность многофункционального онлайн чата с пользователем, а также возможность использования информации от него.

FullContact - внешняя система, благодаря интеграции с которой, можно получить доступную публичную информацию о пользователе.

Разрабатываемая система (BlackBox) - система, которая позволит всем перечисленным выше актерам выполнить свои функции. Она будет подробно рассмотрена в разделе 4.2 (рисунок 4.1).

3.2 Спецификация функциональных требований

Таким образом основной целью данного проекта является создание микросервисной архитектуры, позволяющей легко внедрять и использовать другие программные средства и интегрировать их друг с другом.

В ходе разработки будут реализованы следующие возможности:

- создания ядра событийной трэкинговой системой (event tracking system);
- интеграция с онлайн чатом CRM Salesforce;
- возможность в администриационной панели настраивать модели событий (создания/редактирования/удаления); и возможность просмотра статистики происходящих событий;
- возможность в администриационной просмотра (online/offline) статистики происходящих событий;
- создание сервиса для нахождения публичной информации о пользователе с помощью интеграции с FullContact;
- создание отказоустойчивой микросервисной архитектуры с возможностью горизонтально масштабировать отдельные сервисы;

3.3 Спецификация нефункциональных требованиями

Программное средство должно обладать следующими нефункциональными требованиями:

- работать в любом современном браузере(Chrome 40+, IE9+, Safari 7.7+, Firefox 3.6+);
 - быть легко интегрируемым для клиентов;
 - быть легковесным и иметь как можно меньше накладных расходов для систем клиентов (подгружаемые скрипты должны быть < 10kb);
 - иметь интуитивно понятный интерфейс для просмотра статистик, чтобы новый пользователь мог его освоить за 2-3 дня;
 - иметь возможность легко включаться, выключаться и удаляться из систем клиентов;
 - иметь задержку не более 200ms при обращении к сервисам;
 - уметь возможность работать с большим объемом поступающих входных данных (гигабайтами в день).

4 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

4.1 Архитектура микросервисов

Так как разрабатываемая система должна быть легко расширяема для нового функционала и интеграци, быть высокопроизводительной, отказоустойчивой и приспособлена к работе с большим количеством поступающих данных, то в качестве основной модели положенной в разработку данного программного средства была выбрана микросервисная архитектура.

Архитектурный стиль микросервисов — это подход, при котором единое приложение строится как набор небольших сервисов, каждый из которых работает в собственном процессе и коммуницирует с остальными используя легковесные механизмы, как правило HTTP.

Эти сервисы построены вокруг бизнес-потребностей и развертываются независимо с использованием полностью автоматизированной среды. Существует абсолютный минимум централизованного управления этими сервисами. Сами по себе эти сервисы могут быть написаны на разных языках и использовать разные технологии хранения данных. Таким образом система получается легко расширяема для нового функционала и интеграци.

Благодаря тому, что в микросервисной архитектуре легко горизонтально масштабировать отдельные сервисы, её можно приспосабливать к работе с большим количеством поступающих входных данных и добиться высокой производительности.

Микросервисная архитектура делает большой акцент на мониторинге приложения в режиме реального времени, проверке как технических элементов (например, как много запросов в секунду получает база данных), так и бизнес-метрик (например, как много заказов в минуту получает приложение). Семантический мониторинг может предоставить систему раннего предупреждения проблемных ситуаций, позволяя команде разработке подключиться к исследованию проблемы на самых ранних стадиях. Таким образом, можно добиваться отказоустойчивости.

Микросервисная архитектура позволяет упростить и ускорить процесс релиза, так как не требует пересборки и развертывания всего приложения, как в случае с монолитным приложением. Вместо этого нужно развернуть (redeploy) только те сервисы, которые изменились.

4.2 Архитектура программного средства

Диаграмма взаимодействия сервисов разрабатываемого программного средства представлена на (рисунок 4.1). Актером в данном случае является отдельный микросервис системы, имеющий свою роль в системе. Прецедент – эллипс с надписью, обозначающие какие функции предоставляет тот или иной микросервис.

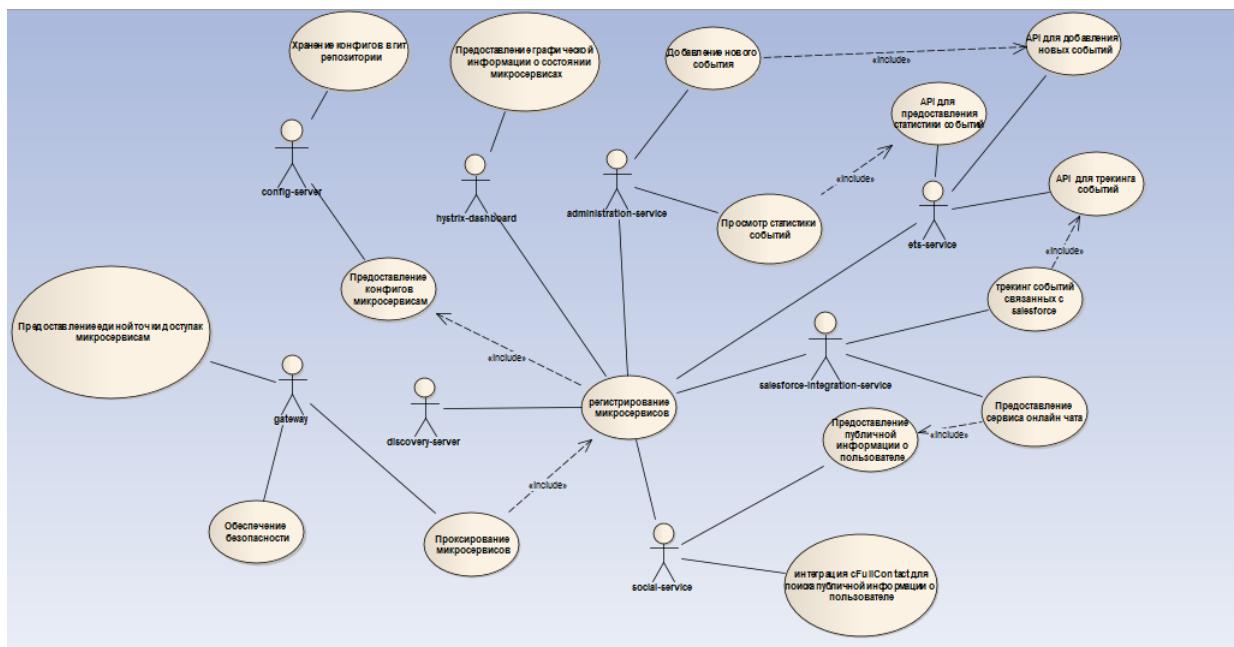


Рисунок 4.1 – Диаграмма взаимодействия сервисов

На основании представленной диаграммы взаимодействия сервисов можно сделать вывод, что в системе будет существовать семь микросервисов:

- gateway;
- config-server;
- discovery-server;
- administration-service;
- salesforce-integration-service;
- social-service;
- ets-service.

Рассмотрим каждого актера с его функциями более подробно:

Discovery-server - этот сервис позволяет микросервисам обнаруживать друг друга и общаться между собой.

Ets-service - основной сервис в котором реализовано ядро с работой над событиями. Оно предоставляет API по созданию различных событий и их отслеживанию.

Administration-client - сервис для администраторов и менеджеров в нем можно настраивать/редактировать/добавлять новые события, следить за их статистикой как за определенный период так и в режиме онлайн.

Salesforce-integration - сервис для интеграции с CRM системой Salesforce. Он также предоставляет онлайн чат для клиентов по различным условиям настраиваемым в Salesforce, а также записывает все необходимые события происходящие с CRM Salesforce с помощью ets-service.

Social-integration - сервис для собираания публичной информации о пользователе, включает в себя интеграцию с FullContact который помогает в сборе информации из доступных социальных сетей, фотографий, географического положения, карьере и около 100 других различных данных о пользователе

Config-server - это сервис для централизованного хранения конфигурации всех микросервисов в отдельном git репозитории.

Hystrix-dashboard - это сервис предоставляет возможность графического мониторинга состояния всех микросервисов.

Gateway - сервис предоставляющий централизованную точку доступа к другим сервисам, а также обеспечивает безопасность для всех остальных сервисов.

5 РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА

При построении данного дипломного проекта необходимо использовать некоторые основные компоненты от Spring Cloud и Netflix OSS, позволяющие отдельно разворачивать микросервисы, а также наладить общение между ними без ручного управления.

В итоге для создания нового экземпляра микросервиса и сбалансирования нагрузки нужно будет выполнить только несколько команд, чтобы начать использовать его без каких-либо ручных настроек.

5.1 Spring Cloud и Netflix OSS

Spring Cloud это достаточно новый продукт из экосистемы spring.io с набором компонент которые удобно использовать для построения микросервисной архитектуры. В значительной степени Spring Cloud базируется на компонентах от Netflix OSS.

Spring Cloud интегрирует компоненты Netflix в экосистему Spring используя автоматическую конфигурацию и конвенцию вместо обычной конфигурации подобно тому, как работает Spring Boot.

Приведенный ниже рисунок 5.1 отображает общие компоненты для реализации отдельных микросервисов:

Operations Component	Netflix, Spring, ELK
Service Discovery server	Netflix Eureka
Dynamic Routing and Load Balancer	Netflix Ribbon
Circuit Breaker	Netflix Hystrix
Monitoring	Netflix Hystrix dashboard and Turbine
Edge Server	Netflix Zuul
Central Configuration server	Spring Cloud Config Server
OAuth 2.0 protected API's	Spring Cloud + Spring Security OAuth2
Centralised log analyses	Logstash, Elasticsearch, Kibana (ELK)

Рисунок 5.1 – Общие компоненты для реализации отдельных микросервисов

Netflix Eureka - Service Discovery Server позволяет микросервисам регистрировать себя во время выполнения, так они появляются в микросервисной архитектуре.

Netflix Ribbon - Динамическая маршрутизация и балансировки нагрузки может быть использована для поиска экземпляра микросервиса для выполнения операции во время выполнения. Netflix Ribbon использует информацию, доступную в Eureka, чтобы найти соответствующие экземпляры служб. Если более чем один экземпляр найден, Netflix Ribbon будет применяться для балансировки нагрузки, чтобы распределить запросы по имеющимся экземплярам. Netflix Ribbon не является отдельным сервисом, но вместо этого используется в качестве встроенного компонента в других микросервисах.

Netflix Zuul - Edge Server Zuul это gateway с внешним миром, он не допускает каким-либо несанкционированным внешним запросам пройти в систему. Edge Server также обеспечивает хорошо известную точку входа к какому-либо микросервису в микросервисной архитектуре. Использование динамики выделенных портов удобно для избежания конфликтов портов и для того чтобы свести к минимуму администрирование. Zuul использует Ribbon для поиска доступных микросервисов и маршрутов и отправляет внешний запрос к соответствующему экземпляру службы.

Netflix Hystrix - Circuit breaker Netflix Hystrix предоставляет возможности автоматического перенаправления вызова если произошли какие-либо проблемы с тем или иным микросервисом. Если микросервис не отвечает (например, из-за тайм-аута или ошибки связи), Hystrix может перенаправить вызов на внутренний метод запасного варианта. Если служба неоднократно не в состоянии ответить, Hystrix будет размыкать цепь вызывая внутренний аварийный метод, не пытаясь вызвать службу на каждом последующем вызове, пока сервис снова не станет доступным. Для того, чтобы определить, является ли сервис снова доступным Hystrix позволяют некоторым запросам попробовать достучаться до сервиса, даже если цепь была разорвана. Hystrix вводится как встроенный компонент в каждый отдельный микросервис.

Netflix Hystrix dashboard and Netflix Turbine - специальное средство для мониторинга статусов сервисов Hystrix. Оно обеспечивает графическое представление информации о состояниях всех сервисов, на основе информации, содержащейся в Eureka.

При реализации данного программного средства были построены следующие микросервисы:

Ets-service

Основной сервис в котором реализовано ядро с работой над событиями. Оно предоставляет API по созданию различных событий и их отслеживанию. Для хранения структуры различных событий используется реалиционная база даны PostgreSQL в которой описаны параметры событий, их

типы, доступные значения, значения по умолчанию, наличие обязательности тех или иных полей и другая информации характеризующая то или иное событие. Для отслеживания же самих событий используется NoSql база данных Mongo. Она лучше всего подходит для этой цели потому что, является быстрой бесструктурной документо-ориентированной базой данных, что позволяют сохранять любые события созданные пользователем, а также изменять их в любое время. Отслеживание события происходит через предоставленное API. Перед тем как оно будет записано, происходит валидация на то что событие корректно, что осуществляется с помощью проверки его структуры. Для валидации же структуры все события подгружаются в Redis (высокопроизводительное нереляционное распределённое хранилище данных).

Administration-client

Сервис разработанный для администраторов и менеджеров в нем можно настраивать/редактировать/добавлять новые события, следить за их статистикой как за определенный период так и в режиме онлайн. Пользовательский интерфейс разработан отдельно на Angular. Также стоит отметить что все API документируются с помощью Swagger, что позволяет frontend разработчикам сгенерировать нужное им API для обращения к построенной системе.

Salesforce-integration

Сервис, предоставляющий интеграцию с CRM системой Salesforce. В котором также разработана интеграция онлайн чата с клиентом, с сохранением соответствующие события с помощью Ets-service.

Social-integration

Микросервис, разработанный для собирания публичной информации о пользователе, включающий в себя интеграцию с FullContact, который помогает в сборе информации из доступных социальных сетей, фотографий, географического положения, карьере и около 100 других различных данных о пользователе

Discovery-server

Этот сервис позволяет микросервисам обнаруживать друг друга и общаться между собой (в данном случае использовалась Eureka).

Config-server

Это сервис, использующий Spring Cloud Config для централизованного хранения конфигурации всех микросервисов в отдельном git репозитории.

Hystrix-dashboard

Во всех микросервисах использовался Hystrix для случаев, когда какой-то из них откажет. Данный же сервис предоставляет возможность графического мониторинга состояния всех микросервисов.

Gateway

Микросервис, использующий Netflix Zuul для предоставления централизованной точки доступа к микросервисам, также с помощью него и Spring Cloud Security настроена безопасность для всех микросервисов.

Для развертывания микросервисов используется docker и docker-compose. С помощью них можно горизонтально масштабировать любой микросервис одной командой.

Например одной командой docker-compose scale ets-service=5, поднимется 5 экземпляров микросервиса ets-service, и с помощью Ribbon другие микросервисы будут балансировать нагрузку по ним при обращении к сервису ets-service.

5.2 Алгоритм сохранения события

Запросы на сохранение события, как и все остальные, сначала посылаются в микросервис gateway. В нем выполняется базовая аутентификация, которая проверяет, чтобы запросы обрабатывались только от клиентов, обладающих на это правами. Далее запрос отправляется с помощью Zuul Proxy в ets microservice, в котором находится логика обработки событий.

В самом же ets микросервисе событие о сохранении сразу же помещается в очередь сообщений RabbitMQ, таким образом все последующие действия выполняются асинхронно, и на стороне пользователя нету задержки, связанной с сохранением событий. Тем временем обработчик событий RabbitMQ, постепенно обрабатывает все прилетевшие в него сообщения. Он же, в свою очередь, сначала пытается распознать к какому из типов моделей событие относится, то или иное сообщений, после чего, если тип найден, сохраняет событие его в MongoDB.

После этого посыпается ещё одно сообщение в RabbitMQ, теперь уже о результатах сохранения события, благодаря чему, все последующие действия опять выполняются асинхронно. Обработчик сообщений предназначенный для обработки результатов сохранения в RabbitMQ, в свою очередь сохраняет результаты статистики и агрегирует их за последнее время по заданным критериям в системе. После чего эти данные отправляются в администрирование модель, через вебсокеты, для того чтобы в ней можно было следить за происходящими событиями на сайте в режиме реального времени.

На рисунке 5.2 изображен алгоритм сохранения события.



Рисунок 5.2 – Алгоритм сохранения события

5.3 Алгоритм классификации событий

Все модели событий, которые принимает наша система, должны быть описаны в реалиционной базе данных (в нашем случае PostgreSQL) и их можно добавлять, удалять и изменять в любое время.

Описание каждой модели включает в себя тип и набор полей, которые в неё входят. А каждое поле, в свою очередь, содержит информацию, к какому типу оно должно принадлежать(Date, Double, String, Long, List) и то, обязательно оно или нет. Алгоритм классификации событий определяет, к каким моделям событий можно соотнести пришедшее на вход событие, чтобы тип совпадал и все валидации проходили. Первое что нужно алгоритму, это все модели данных, но так как все время обращаться в реалиционную базу за ними было бы очень дорого и неэффективно, все модели подгружаются в *in memory* базу Redis. Для того, чтобы данные в ней были все время актуальны, на ней настроен TTL, и поэтому через n-ое время все модели событий обновляются и всё время являются актуальными.

Сначала алгоритм ищет все модели событий у которых тип соответствует полю типа входного события, далее он идет по всем полям модели события, и проверяет, что пришедшее событие подходит под его модель: проверяет все обязательные поля и типы полей. Таким образом данный алгоритм находит все модели которые подходят для пришедшего события, а также валидирует их.

На рисунке 5.3 приведена схема алгоритма классификации события.

5.4 Схема интеграции с онлайн чатом Salesforce

Рассмотрим схему алгоритма интеграции с онлайн-чатом Salesforce.

Для интеграции с онлайн чатом Salesforce нужны стандартные объекты Salesforce, такие как LiveChatButton, DeploymentID и OrganizationID. Они включают в себя большое количество опций прямо из коробки Salesforce. Их описание можно найти в соответствующей документации по SF. Отметим только, что LiveChatButton связывается с DeploymentID, и SF при подгрузке пытается подгрузить все LiveChatButton по соответствующим условиям, описаным в SF. Также на LiveChatButton есть связь со SkillID, по которому тоже можно фильтровать то, какие из кнопок должны подгружаться.

Дополнительно в SF были созданы собственные объекты Site и RelativeURL. В первом размещается общая информация о сайте и главное его домен и сниппет. Во втором есть связь с сайтом, DeploymentID, SkillID, Callback и относительная страница, но которую эти настройки должны применяться. После того, как SF сконфигурирован, в него могут заходить агенты и приступать к своей работе (общению с пользователями сайта, собираанию статистик, настройке новых страниц, в Salesforce CRM можно делать очень многое).

Для интеграции с сервисом, клиент должен разместить сниппет, который он может найти в Salesforce CRM. После того как сниппет подгрузится на странице пользователя, он делает запрос на извлечение данных необходимых для подгрузки онлайн чата. После того как сниппет был отправлен и аутентифицирован, запрос перенаправляется в salesforce-integration microservice.

В нем определяется, с какой страницы был произведен запрос, и дальше ищется в базе синтегрированной с SF, к какой связке (Site, RelativeURL) она больше всего подходит. Дальше из этой связки достаются нужные OrganizationID DeploymentId, SkillID и Callback. Следующий шаг - это найти все подходящие LiveChatButton. Они определяются по найденным DeploymentID и SkillID. После того, как вся нужная информация собрана (OrganizationID, DeploymentId, SkillID, Callback и список LiveChatButton) она передается в браузер клиенту, и в нём подгружается Salesforce онлайн-чат по пришедшим параметрам. После того, как он подгрузился, выполняется дополнительный callback, переданный из SF, который позволяет выполнить любые дополнительные действия, нужные в тех или иных случаях.

На рисунке 5.4 приведена схема интеграции.

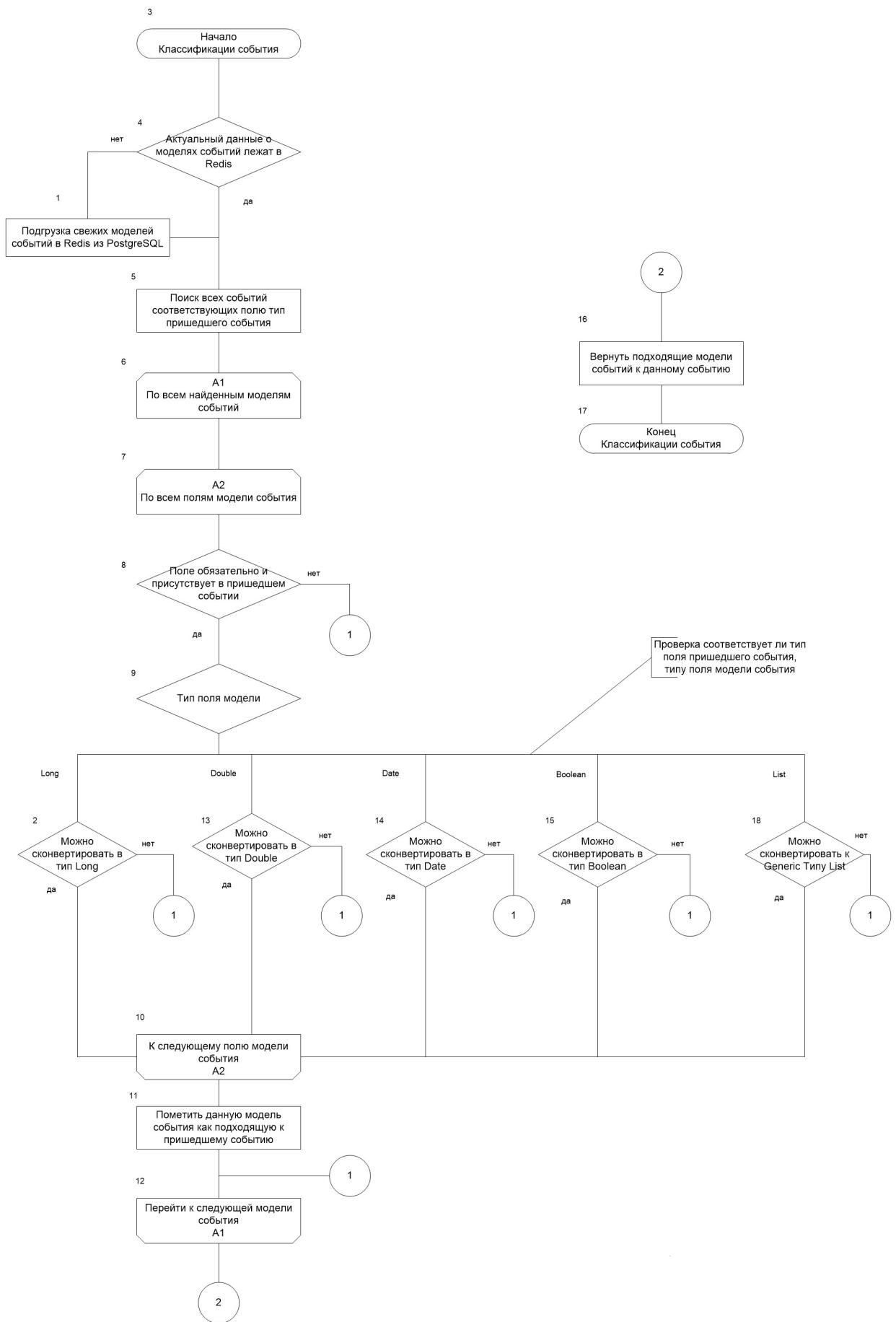


Рисунок 5.3 – Алгоритм классификации события

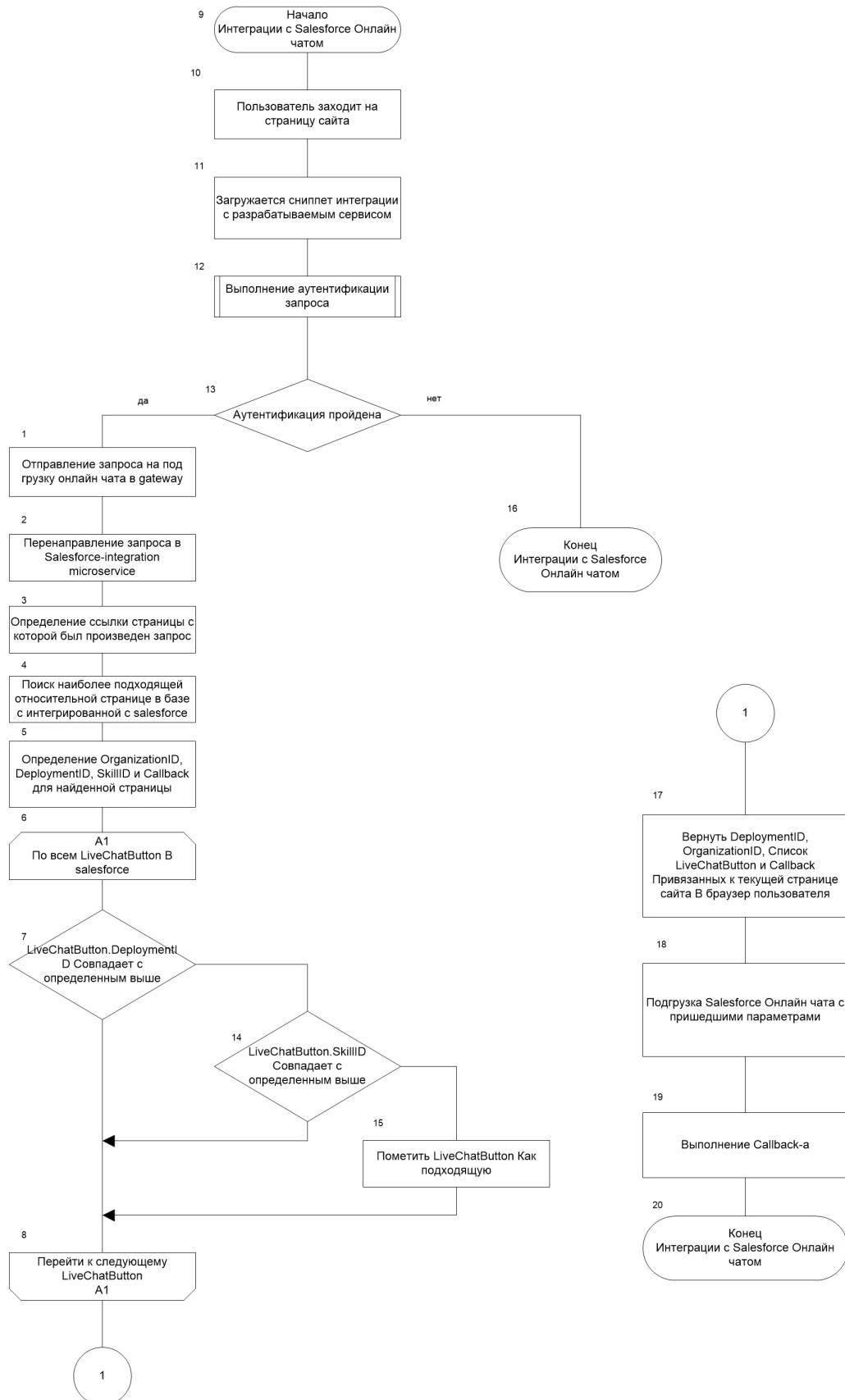


Рисунок 5.4 – Схема интеграции с Salesforce online chat

6 ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ

6.1 Тестирование администрирования панели

Для оценки правильности работы функциональных требований в администрировании панели было проведено тестирование, включающие тест кейсы представленные в таблице 6.1

Таблица 6.1 – Тестирование микросервиса администрирования панели

Название тест-кейса и его описание	Ожидаемый результат	Полученный результат
Авторизация в администрирование панель – Зайти на страницу входа в аккаунт администрирования панели – Ввести логин и пароль – Нажать кнопку Login	<ul style="list-style-type: none">– Отображается страница входа в аккаунт администрирования панели– Необходимые поля доступны для заполнения– Вход в администрирование панель	<ul style="list-style-type: none">– Отображается страница входа в аккаунт администрирования панели– Необходимые поля доступны для заполнения– Вход в администрирование панель
Создание нового события – Перейти на страницу создания нового события – Ввести обязательные поля – Нажать кнопку Create	<ul style="list-style-type: none">– Отображается страница создания нового события– Необходимые поля доступны для заполнения– Событие успешно создано	<ul style="list-style-type: none">– Отображается страница создания нового события– Необходимые поля доступны для заполнения– Событие успешно создано

<p>Добавление поля для события</p> <ul style="list-style-type: none"> – Выбрать событие – Нажать кнопку Add Prop – Ввести обязательные поля – Нажать кнопку Add 	<ul style="list-style-type: none"> – Отображается страница события – Отображается попап добавления поля события – Необходимые поля доступны для заполнения – Поле успешно добавлено в событие 	<ul style="list-style-type: none"> – Отображается страница события – Отображается попап добавления поля события – Необходимые поля доступны для заполнения – Поле успешно добавлено в событие
<p>Просмотр онлайн статистики поступление событий</p> <ul style="list-style-type: none"> – Зайти в административную панель – Перейти на страницу статистики 	<ul style="list-style-type: none"> – Отображается главной страницы – Отображается график текущих поступающими событиями, который изменять с течение времени без перезагрузки страницы 	<ul style="list-style-type: none"> – Отображается главной страницы – Отображается график текущих поступающими событиями, который изменять с течение времени без перезагрузки страницы
<p>Просмотр статистики поступивших событий за промежуток времени</p> <ul style="list-style-type: none"> – Зайти в административную панель – Перейти на страницу статистики – Выбрать интервал времени – Нажать кнопку Show 	<ul style="list-style-type: none"> – Отображается главной страницы – Отображается график текущих поступающими событиями, который изменять с течение времени без перезагрузки страницы – Необходимые поля доступны для заполнения – Отображается график поступивших событий за выбранный промежуток 	<ul style="list-style-type: none"> – Отображается главной страницы – Отображается график текущих поступающими событиями, который изменять с течение времени без перезагрузки страницы – Необходимые поля доступны для заполнения – Отображается график поступивших событий за выбранный промежуток

Таким образом результат тестирования подтверждает, что микросервис ответственный за администрирование панель работает корректно с установленными требованиями.

6.2 Тестирование интеграции с Salesforce CRM

Для оценки правильности работы функциональных требований связанных с интеграцией salesforce было проведено тестирование, включающие тест кейсы представленные в таблице 6.2

Таблица 6.2 – Тестирование микросервиса интеграции с Salesforce CRM

Название тест-кейса и его описание	Ожидаемый результат	Полученный результат
Тестирование настроек онлайн чата в sf – Агент заходит в Salesforce CRM – Агент настраивает параметры в salesforce для онлайн чата (специфика sf) – Агент настраивает страницу сайта на которой должен появляться чат – Агент заходит в Salesforce Console	– Отображается главной страница Salesforce CRM – Параметры доступны для настройки – Необходимые поля доступны для заполнения – Отображается Salesforce Console доступна	– Отображается главной страница Salesforce CRM – Параметры доступны для настройки – Необходимые поля доступны для заполнения – Отображается Salesforce Console доступна

<p>Тестирование онлайн чата</p> <ul style="list-style-type: none"> – Агент ставит в Salesforce Console себе статус онлайн – Пользователь заходит на страницу сайта, на которой настроен онлайн чат – Выполняются условия по которым должен показаться онлайн чат – Пользователь нажимает кнопку Start Chat – Агент нажимает кнопку принять чат – Агент отправляет сообщение – Пользователь отправляет сообщение – Пользователь закрывает чат 	<ul style="list-style-type: none"> – Агенту простовляется статус онлайн – Страница сайта корректно загружается – На странице всплывает окошко для онлайн чата – Открывается окно ожидание онлайн чата с агентом и у агента в Salesforce Console отображается входящий вызов – Открывается чат с агентом – Пользователь видет сообщение – Агент видет сообщение – Чат успешно закрывается 	<ul style="list-style-type: none"> – Агенту простовляется статус онлайн – Страница сайта корректно загружается – На странице всплывает окошко для онлайн чата – Открывается окно ожидание онлайн чата с агентом и у агента в Salesforce Console отображается входящий вызов – Открывается чат с агентом – Пользователь видет сообщение – Агент видет сообщение – Чат успешно закрывается
---	--	--

Таким образом результат тестирования подтверждает, что микросервис ответственный за интеграцию с salesforce работает корректно с установленными требованиями.

7 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

7.1 Руководство по разворачиванию приложения

Приложения разрабатывалось с помощью package manager-a gradle, в котором прописаны все необходимые сторонние библиотеки, таким образом все нужный зависимости для проекта скачиваются автоматически. Зависимости для UI собирались с помощь package manager-a npm, поэтому они так же скачиваются автоматически, при сборке проекта.

Приложение использует следующие базы данных и сервисы:

- PostgreSQL 9.3
- MongoDB 3.0
- Redis 3.2
- RabbitMQ 3.6
- Salesforce
- FullContact

Всю конфигурацию для подключения баз данных и внешних сервисов, нужно прописывать в соответствующих файлах в git репозитории конфигурации:

- Конфигурацию для подключению PostgreSQL, MongoDB, Redis, RabbitMQ в файле ets-service.yml
- Конфигурацию для подключению Salesforce и с интегрированой с ним базой в файле salesforce-integration.yml
 - Конфигурацию для подключения FullContact в файле social-integration.yml
 - После того как соответствующую конфигурацию будет прописана, нужно выполнить стандартные команды git-а: git commit -a и git push.

После этого, чтобы собрать приложение нужно выполнить команду gradle build.

Если нужно запустить отдельный микросервис то его можно запустить соответствующей командой gradle :microservice-name:bootRun, где microservice-name имя микросервиса.

Для простоты разворачивания всех микросервисов сразу, был написан docker-compose.yml файл, с помощью которого одной командой docker-compose up, можно запустить все микросервисы сразу в docker контейнерах. Кроме того, он дает возможность легко горизонтально масштабировать отдельные микросервисы.

Например: командой docker-compose scale ets-service=5, поднимется 5 экземпляров микросервиса ets-service, и с помощью Ribbon другие микро-

сервисы будут балансировать нагрузку по ним при обращении к сервису ets-service.

Для просмотра информации о развернутых серверах и их состояниях, можно зайти на host:8761, где host - это адрес где развернут микросервис discovery-client (рисунок 7.1).

The screenshot shows the Spring Eureka interface at localhost:8761. It includes sections for System Status, DS Replicas, and General Info, displaying various configuration and service registration details.

System Status

Environment	test	Current time	2016-05-10T20:24:28 +0300
Data center	default	Uptime	04:17
		Lease expiration enabled	true
		Renews threshold	6
		Renews (last min)	12

DS Replicas

localhost

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
ADMINISTRATION-CLIENT	n/a (1)	(1)	UP (1) - 192.168.99.1:administration-client:7000
ETS-SERVICE	n/a (1)	(1)	UP (1) - 192.168.99.1:ets-service:8020
SALESFORCE-INTEGRATION	n/a (1)	(1)	UP (1) - localhost

General Info

Name	Value
total-avail-memory	378mb
environment	test

Рисунок 7.1 – Discovery client interface

7.2 Руководство для администраторов и менеджеров

В рамках данного проекта была также разработана администрирование панель, в которой администраторы и менеджеры могут создавать/удалять/изменять и настраивать модели событий, которые будет поддерживать система, а так же возможность просмотра реал-тайм статистики происходящих событий. В рамках данной работы, постройка сложного UI не предполагалась, а наоборот цель была построить как можно более минималистический интерфейс, для проверки концепции архитектуры данной системы.

На рисунке 7.2, можно увидеть интерфейс на котором можно настраивать модели события под свои нужды.

На рисунке 7.3 показано как выглядит интерфейс добавления и изменения того или иного поля в модели события.

The screenshot shows a web-based administration interface for managing event models. At the top, there's a header bar with various links like 'SF', 'Spring', 'Be good if read', etc. Below the header, the title 'Event Models' is displayed. A green button labeled 'Create Model' is visible. The main content area contains two tables:

Event Type	Event Props						Actions
	Id	Name	Type	Required	Default	Action	
test_event_1	24	Test Long Property	LONG	true	0	<button>Delete</button>	
	27	Test Date Property	DATE	true		<button>Delete</button>	
	25	Test Double Property	DOUBLE	true	0.0	<button>Delete</button>	
	26	Test String Property	STRING	true		<button>Delete</button>	
	28	Test List Property	LIST	true	yes/no	<button>Delete</button>	

Event Type	Event Props						Actions
	Id	Name	Type	Required	Default	Action	
order_created	30	orderId	LONG	true		<button>Delete</button>	
	29	customerId	LONG	true		<button>Delete</button>	
	32	info	LONG	true		<button>Delete</button>	
	31	purchase	STRING	true		<button>Delete</button>	

Below the tables, there are buttons for 'Add prop' and 'Delete Event'.

Рисунок 7.2 – Интерфейс настройки моделей событий

Интерфейс просмотра статистики изображен на рисунке 7.4. Так же на нем существует возможность просмотра статистики за определенный период. Для этого нужно всего ли выбрать соответствующий период.

7.3 Руководство по интеграции с онлайн чатом

На староне salesforce есть множество настроек и критериев, по которым можно настроить онлайн чат под свои нужды. И в данный проект не подразумевает описание того как работает Salesforce. Подразумевается, что агенты которые будут общаться с посетителями через онлайн чат Salesforce, знают его и умеют с ним работать. Со стороны агента чат будет как на рисунке 7.5, также там предоставляются объекты Contact и Case в которых он заполняет, всё необходимую информацию собранную у клиента.

На староне посетителя, чат выглядит как на рисунке 7.6.

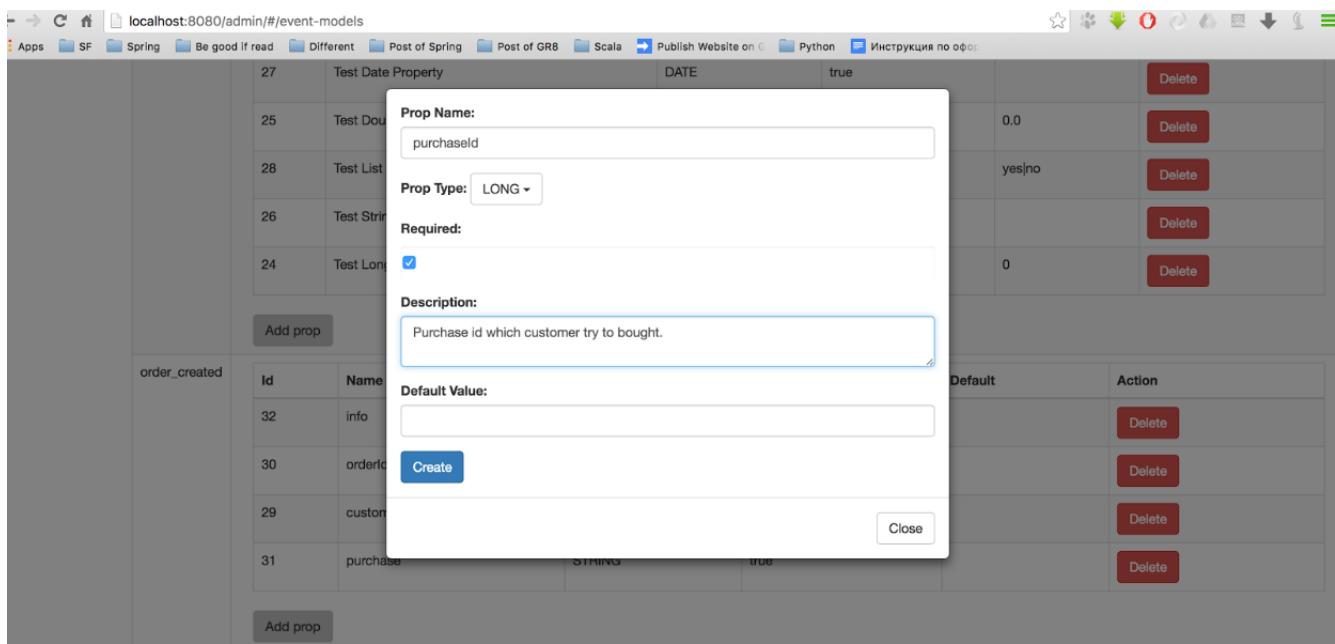


Рисунок 7.3 – Интерфейс настройки поля модели события

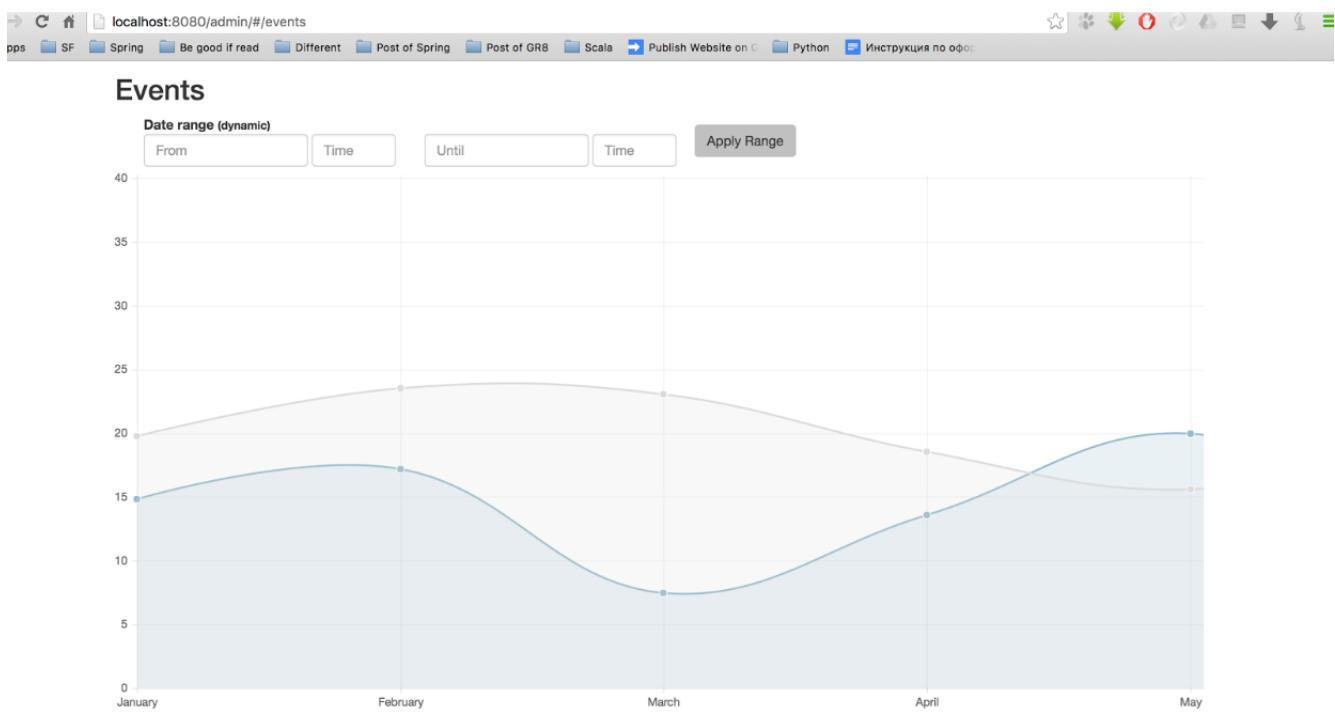


Рисунок 7.4 – Интерфейс просмотра статистики

The screenshot shows the Salesforce online chat interface. On the left, there is a sidebar with a list of messages from a visitor. The main area shows a 'New Contact' form with fields for contact information like name, phone, and address. The 'Contact Information' section includes fields for Contact Owner, First Name (Ivanov), Last Name (Ivanov), Account Name, Title, Department (IT), Birthdate (5/2/2016), Reports To, Lead Source, Phone (327 837-7483), Home Phone (37827484372), Mobile (232 323-4342), Other Phone, Fax, Email (ivanov.sutak@gmail.com), Assistant, Asst. Phone, and Address information for Mailing Street, City (Kiev), State/Province, Zip/Postal Code, and Country.

Рисунок 7.5 – Salesforce online chat

The screenshot shows a client-side chat interface. The user (Me) sends a message: "Yes, I want to know how i can to repair my macbook". The support agent (aleh s) responds with: "its not working", "something bad happend with him", "nothing help", and "Im dissapointed". The user also sends messages: "Hello aleh!", "Can I help you?", "Our Supported Group help you", "all would be fun", "Zachtile Diplom", and "PLiz". The support agent ends the conversation with "Ok".

Рисунок 7.6 – Client side chat

8 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ ПС

Целью дипломного проектирования является создание информационно-аналитического программного средства для взаимодействия клиента с сайтом с целью увеличения продуктивности последнего. Данное ПС реализует интеграцию сайта с различными CRM-системами, создания профиля пользователя, а также собирание статистики и действий произведенных им на сайте. Основными достоинствами программного средства являются: система отслеживания событий приспособленная для интеграции с такими внешними сервисами, как ContactInfo, Salesforce CRM; микросервисная архитектура, позволяющая горизонтально масштабировать отдельные сервисы и легко добавлять новые интеграции с внешними источниками.

В данном разделе рассмотрим экономическую эффективность программного средства. Для оценки экономической эффективности разработанного программного средства необходимо рассчитать смету затрат на разработку, цену и прибыль продажи одной программной системы.

Программный комплекс относится к 3-й группе сложности. Категория новизны продукта – «Б».

Расчеты выполнены на основе методического пособия [?].

8.1 Расчёт сметы затрат и цены программного продукта

Исходные данные для разрабатываемого проекта указаны в таблице 8.1.

Таблица 8.1 – Исходные данные

Наименование	Условное обозначение	Значение
Категория сложности		3
Коэффициент сложности, ед.	K_c	1,12
Степень использования при разработке стандартных модулей, ед.	K_t	0,6
Коэффициент новизны, ед.	K_n	0,9
Годовой эффективный фонд времени, дн.	$\Phi_{\text{эфф}}$	231
Продолжительность рабочего дня, ч.	T_u	8
Месячная тарифная ставка первого разряда, Вр	T_{M_1}	450 000
Коэффициент премирования, ед.	K	1,5

Норматив дополнительной заработной платы, ед.	H_d	20
Норматив отчислений в ФСЗН, %	H_{cz}	34
Норматив отчислений в Белгосстрах, %	H_c	34
Норматив командировочных расходов, %	H_k	15
Норматив прочих затрат, %	H_{pz}	20
Норматив накладных расходов, %	H_{ph}	100
Прогнозируемый уровень рентабельности, %	Y_{pp}	35
Норматив НДС, %	H_{dc}	20
Норматив налога на прибыль, %	H_p	18
Норматив расхода материалов, %	H_{mz}	3
Норматив расхода машинного времени, ч.	H_{mb}	15
Цена одного часа машинного времени, Br	H_{mb}	25 000
Норматив расходов на сопровождение и адаптацию ПО, %	H_{psa}	30

На основании сметы затрат и анализа рынка ПО определяется плановая отпускаемая цена. Для составления сметы затрат на создание ПО необходима предварительная оценка трудоемкости ПО и его объёма. Расчет объёма программного продукта (количества строк исходного кода) предполагает определение типа программного обеспечения, всестороннее техническое обоснование функций ПО и определение объёма каждой функций. Согласно классификации типов программного обеспечения [? , с. 59, приложение 1], разрабатываемое ПО с наименьшей ошибкой можно классифицировать как ПО методо-ориентированных расчетов.

Общий объём программного продукта определяется исходя из количества и объёма функций, реализованных в программе:

$$V_o = \sum_{i=1}^n V_i, \quad (8.1)$$

где V_i — объём отдельной функции ПО, LoC;
 n — общее число функций.

На стадии технико-экономического обоснования проекта рассчитать точный объём функций невозможно. Вместо вычисления точного объёма функций применяются приблизительные оценки на основе данных по аналогичным проектам или по нормативам [? , с. 61, приложение 2], которые приняты в

организации.

Таблица 8.2 – Перечень и объём функций программного модуля

№ функции	Наименование (содержание)	Объём функции, LoC	
		по каталогу (V_i)	уточненный (V_i^y)
101	Организация ввода информации	620	450
405	Система настройки ПО	1650	1500
502	Монитор система (управление работой комплекса ПО)	1100	900
301	Интерфейс к базе данных моделей и данных аутентификации	790	650
305	Защита от несанкционированного доступа к базе	820	400
501	Монитор ПО (управление работой компонентов)	1240	1010
506	Обработка исключительных ситуаций	620	400
507	Обеспечение интерфейса между компонентами	970	680
612	Аутентификация на основе сравнения реальных данных и математической модели	1100	800
Итог		9710	6390

Перечень и объём функций программного модуля перечислен в таблице 8.2. По приведенным данным уточненный объём некоторых функций изменился, и общий уточненный объём ПО $V_y = 6390$ LoC .

8.2 Расчёт трудоемкости

На основании общего объема ПО определяется нормативная трудоемкость (T_n) с учетом сложности ПО. Для ПО 3-ой группы сложности, к которой относится разрабатываемый программный продукт, нормативная трудоемкость составит $T_n = 144$ чел./дн.

Нормативная трудоемкость служит основой для оценки общей трудо-

емкости T_o . Используем формулу (8.2) для оценки общей трудоемкости для небольших проектов:

$$T_o = T_h \cdot K_c \cdot K_t \cdot K_n, \quad (8.2)$$

где K_c — коэффициент, учитывающий сложность ПО;

K_t — поправочный коэффициент, учитывающий степень использования при разработке стандартных модулей;

K_n — коэффициент, учитывающий степень новизны ПО.

Дополнительные затраты труда на разработку ПО учитываются через коэффициент сложности, который вычисляется по формуле

$$K_c = 1 + \sum_{i=1}^n K_i, \quad (8.3)$$

где K_i — коэффициент, соответствующий степени повышения сложности ПО за счет конкретной характеристики;

n — количество учитываемых характеристик.

Наличие двух характеристик сложности позволяет [?, с. 66, приложение 4, таблица П.4.2] вычислить коэффициент сложности

$$K_c = 1 + 0,12 = 1,12. \quad (8.4)$$

Разрабатываемое ПО использует стандартные компоненты. Согласно справочным данным [?, с. 68, приложение 4, таблица П.4.5] коэффициент использования стандартных модулей для разрабатываемого приложения $K_t = 0,6$.

Согласно справочным данным [?, с. 67, приложение 4, таблица П.4.4], коэффициент новизны для разрабатываемого ПО $K_n = 0,9$.

Подставив приведенные выше коэффициенты для разрабатываемого ПО в формулу (8.2) получим общую трудоемкость разработки

$$T_o = 144 \times 1,12 \times 0,6 \times 0,9 \approx 87 \text{ чел./дн.} \quad (8.5)$$

На основе общей трудоемкости и требуемых сроков реализации проекта вычисляется плановое количество исполнителей. Численность исполнителей проекта рассчитывается по формуле:

$$\Psi_p = \frac{T_o}{T_p \cdot \Phi_{\text{эфф}}}, \quad (8.6)$$

где T_o — общая трудоемкость разработки проекта, чел./дн.;
 $\Phi_{\text{эф}}$ — эффективный фонд времени работы одного работника в течение года, дн.;
 T_p — срок разработки проекта, лет.

Эффективный фонд времени работы одного разработчика вычисляется по формуле

$$\Phi_{\text{эф}} = \Delta_g - \Delta_n - \Delta_v - \Delta_o, \quad (8.7)$$

где Δ_g — количество дней в году, дн.;
 Δ_n — количество праздничных дней в году, не совпадающих с выходными днями, дн.;
 Δ_v — количество выходных дней в году, дн.;
 Δ_o — количество дней отпуска, дн.

Согласно данным, приведенным в производственном календаре для пятидневной рабочей недели в 2016 году для Беларуси [?], фонд рабочего времени составит

$$\Phi_{\text{эф}} = 366 - 6 - 105 - 24 = 231 \text{ дн.} \quad (8.8)$$

Учитывая срок разработки проекта $T_p = 3 \text{ мес.} = 0,25 \text{ года}$, общую трудоемкость и фонд эффективного времени одного работника, вычисленные ранее, можем рассчитать численность исполнителей проекта

$$Ч_p = \frac{87}{0,25 \times 231} \approx 2 \text{ рабочих.} \quad (8.9)$$

Вычисленные оценки показывают, что для выполнения запланированного проекта в указанные сроки необходимо 2 рабочих.

8.3 Расчёт заработной платы исполнителей

Информация о работниках перечислена в таблице 8.3.

Таблица 8.3 – Работники, занятые в проекте

Исполнители	Разряд	Тарифный коэффициент	Чел./дн. занятости
Программист I-категории	14	3,25	43
Ведущий программист	15	3,48	44

Месячная тарифная ставка одного работника вычисляется по формуле

$$T_q = \frac{T_{M_1} \cdot T_k}{\Phi_p}, \quad (8.10)$$

где T_{M_1} — месячная тарифная ставка 1-го разряда, Br;
 T_k — тарифный коэффициент, соответствующий установленному тарифному разряду;
 Φ_p — среднемесячная норма рабочего времени, час.

Подставив данные из таблицы 8.3 в формулу (8.10), приняв значение тарифной ставки 1-го разряда $T_{M_1} = 450\,000$ Br и среднемесячную норму рабочего времени $\Phi_p = 160$ часов получаем

$$T_q^{\text{прогр. I-разр.}} = \frac{450\,000 \times 3,25}{160} = 9141 \text{ Br/час}; \quad (8.11)$$

$$T_q^{\text{вед. прогр.}} = \frac{450\,000 \times 3,48}{160} = 9788 \text{ Br/час}. \quad (8.12)$$

Основная заработная плата исполнителей на конкретное ПО рассчитывается по формуле

$$Z_o = \sum_{i=1}^n T_q^i \cdot T_q \cdot \Phi_p \cdot K, \quad (8.13)$$

где T_q^i — часовая тарифная ставка i -го исполнителя, Br/час;
 T_q — количество часов работы в день, час;
 Φ_p — плановый фонд рабочего времени i -го исполнителя, дн.;
 K — коэффициент премирования.

Подставив ранее вычисленные значения и данные из таблицы 8.3 в формулу (8.13) и приняв коэффициент премирования $K = 1,5$ получим

$$Z_o = (9141 \times 43 + 9788 \times 44) \times 8 \times 1,5 = 9\,884\,820 \text{ Br}. \quad (8.14)$$

Дополнительная заработная плата включает выплаты предусмотренные законодательством от труда и определяется по нормативу в процентах от основной заработной платы

$$Z_d = \frac{Z_o \cdot H_d}{100\%}, \quad (8.15)$$

где H_d — норматив дополнительной заработной платы, %.

Приняв норматив дополнительной заработной платы $H_d = 20\%$ и подставив известные данные в формулу (8.15) получим

$$Z_d = \frac{9\ 884\ 820 \times 20\%}{100\%} \approx 1\ 976\ 964 \text{ Br.} \quad (8.16)$$

8.4 Расчёт расходов и прогнозируемой цены ПО

Расчеты общей суммы расходов и прогнозируемой цены ПО, а также его себестоимости сведены в таблицу 8.4.

Таблица 8.4 – Расчет себестоимости и отпускной цены ПО

Наименование статей	Норматив, %	Методика расчета	Значение, руб.
Отчисления в фонд социальной защиты и обязательного страхования	$H_{cz} = 34$	$Z_{cz} = (Z_o + Z_d) \cdot H_{cz}/100$	4 033 007
Отчисления в Белгосстрах	$H_c = 0,7$	$Z_{cz} = (Z_o + Z_d) \cdot H_c/100$	83 032
Материалы и комплектующие	$H_{mz} = 3$	$M = Z_o \cdot H_{mz}/100$	296 545
Машинное время		$P_m = \Pi_m \cdot V_o/100 \cdot H_{mb}$ $H_{mb} = 15$ машино-часов $\Pi_m = 25\ 000 \text{ Br}$	23 962 500
Расходы на научные командировки	$H_k = 15$	$P_k = Z_o \cdot H_k/100$	1 482 723
Прочие прямые расходы	$H_{pz} = 20$	$\Pi_3 = Z_o \cdot H_{pz}/100$	1 976 964
Накладные расходы	$H_{ph} = 100$	$P_h = Z_o \cdot H_{ph}/100$	9 884 820
Общая сумма расходов по смете		$C_p = Z_o + Z_d + Z_{cz} + M + P_m + P_k + \Pi_3 + P_h$	72 117 899
Сопровождение и адаптация ПО	$H_{pca} = 30$	$P_{ca} = C_p \cdot H_{pca}/100$	16 642 592

Полная себестоимость ПО		$C_{\text{пп}} = C_p + P_{\text{са}}$	72 117 899
Прогнозируемая прибыль	$Y_{\text{пп}} = 35$	$\Pi_c = C_{\text{пп}} \cdot Y_{\text{пп}} / 100$	25 241 265
Прогнозируемая цена без налогов		$\Pi_{\text{пп}} = C_{\text{пп}} + \Pi_c$	97 359 164
Налог на добавленную стоимость	$H_{\text{дс}} = 20$	$\text{НДС} = (\Pi_{\text{пп}} + O_{\text{мр}}) \cdot H_{\text{дс}} / 100$	20 262 053
Прогнозируемая отпускная цена		$\Pi_o = \Pi_{\text{пп}} + O_{\text{мр}} + \text{НДС}$	121 572 317

8.5 Расчёт экономической эффективности у разработчика

Важная задача при выборе проекта для финансирования это расчет экономической эффективности проектов и выбор наиболее выгодного проекта. Разрабатываемое ПО является заказным, т.е. разрабатывается для одного заказчика на заказ. На основании анализа рыночных условий и договоренности с заказчиком об отпускной цене прогнозируемая рентабельность проекта составит $Y_{\text{пп}} = 35\%$.

Чистую прибыль от реализации проекта можно рассчитать по формуле

$$\Pi_q = \Pi_c \cdot \left(1 - \frac{H_{\text{пп}}}{100\%} \right), \quad (8.17)$$

где $H_{\text{пп}}$ — величина налога на прибыль, %.

Приняв значение налога на прибыль $H_{\text{пп}} = 18\%$ и подставив известные данные в формулу (8.17) получаем чистую прибыль

$$\Pi_q = 25 241 265 \times \left(1 - \frac{18\%}{100\%} \right) = 20 697 837 \text{ Br}. \quad (8.18)$$

Программное обеспечение разрабатывалось для одного заказчика в связи с этим экономическим эффектом разработчика будет являться чистая прибыль от реализации Π_q . Рассчитанные данные приведены в таблице 8.5.

Таблица 8.5 – Рассчитанные данные

Наименование	Условное обозначение	Значение
Нормативная трудоемкость, чел./дн.	T_n	144
Общая трудоемкость разработки, чел./дн.	T_o	87
Численность исполнителей, чел.	$Ч_p$	2
Часовая тарифная ставка программиста I-разряда, Br/ч.	$T_q^{\text{прогр. I-разр.}}$	9141
Часовая тарифная ставка ведущего программиста, Br/ч.	$T_q^{\text{вед. прогр.}}$	9788
Основная заработка плата, Br	Z_o	9 884 820
Дополнительная заработка плата, Br	Z_d	1 976 964
Отчисления в фонд социальной защиты, Br	Z_{cz}	4 033 007
Затраты на материалы, Br	M	296 545
Расходы на машинное время, Br	P_m	23 962 500
Расходы на командировки, Br	P_k	1 482 723
Прочие затраты, Br	P_3	1 976 964
Накладные расходы, Br	P_h	9 884 820
Общая сумма расходов по смете, Br	C_p	55 475 307
Расходы на сопровождение и адаптацию, Br	P_{ca}	16 642 592
Полная себестоимость, Br	C_n	72 117 899
Прогнозируемая прибыль, Br	P_c	25 241 265
НДС, Br	НДС	20 262 053
Прогнозируемая отпускная цена ПО, Br	Z_o	121 572 317
Чистая прибыль, Br	P_q	20 697 837

8.6 Выводы по технико-экономическому обоснованию

Программное средство разрабатывалось для одного заказчика и в связи с этим экономическим эффектом разработчика будет являться чистая прибыль от реализации P_q . Рассчитанные данные приведены в таблице 8.5. Таким образом, было произведено технико-экономическое обоснование разрабатываемого проекта, составлена смета затрат и рассчитана прогнозируемая прибыль, а также показана экономическая целесообразность разработки.

Информационно-аналитической программное средство для взаимодействия клиента с сайтом является выгодным программным продуктом. Про-

гнозируемая отпускная цена (Π_o) составляет 121 572 317 рублей. Таким образом, данная разработка является экономически целесообразной. Чистая прибыль от реализации ПС ($\Pi_u = 20\,697\,837$ рублей) представляет собой экономический эффект от создания нового программного средства.

ЗАКЛЮЧЕНИЕ

В результате данного дипломного проекта было создано программного средство помогающие сайту лучше взаимодействовать с пользователем. Разработана система по отслеживанию событий, обеспечивающие более легкую интеграцию и взаимодействие между сторонними веб сервисами. Также была разработана интеграция с CRM системой Salesforce обеспечивающая онлайн чат с пользователями по заданным критериям и внешней системой FullContact для получения публичной информации о пользователе.

В следствии работы над данным проектом, были получены знания, в области разработки легко расширяемой, высокопроизводительной, отказоустойчивой системы на основе микросервисной архитектуры.

В результате работы выяснилось, что система работает, эффективна и имеет право на существование.