МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное образовательное
учреждение высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
**Высшая школа электроники и компьютерных наук
Кафедра системного программирования**

РАБОТА ПРОВЕРЕНА

Рецензент
Технический директор OrangeApps

_____ А.В. Гуйдо

“___”_____ 2017 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой,
д.ф.-м.н., профессор

_____ Л.Б. Соколинский

“___”_____ 2017 г.

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ СБОРА СТАТИСТИКИ ПО ИСПОЛЬЗОВАНИЮ МОБИЛЬНЫХ ИГР

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.04.02.2017.115-073.ВКР

Научный руководитель
кандидат физ.-мат. наук, доцент
кафедры СП
_____С.А. Иванов

Автор работы,
студент группы КЭ-217
_____М.С. Туприков

Ученый секретарь
(нормоконтролер)

_____ О.Н. Иванова

“___”_____ 2017 г.

Челябинск-2017

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

Во всем мире возрастает интерес к сбору статистической информации использования мобильных игровых приложений. Особенно большую роль на данный момент статистика влияет в бизнесе, помогает в борьбе за старых и новых клиентов. В области мобильных интерактивных приложений, где компании соревнуются за каждого пользователя, статистика может помочь лучше понять, как нужно улучшить свое приложение, как пользователи используют приложение и что они ожидают.

Сбор нужной статистики помогает узнать, как пользователи реагируют на каждое нововведение в приложении и как в целом пользуются приложением. Объективное понимание, в какую сторону необходимо улучшать приложение для того, чтобы пользователям было удобнее и приятнее пользоваться им, напрямую влияет на получаемую прибыль с каждого пользователя (ARPU).

По статистике [14] за последние пару лет доля мобильного трафика выросла в огромное количество раз. Для мобильного трафика в разных сферах может доходить до 75 %. Количество устройств и их доступность привела к тому, что количество денежного оборота в мобильном сегменте выросла в несколько раз по сравнению с прошлым годом. С ростом прибыльности выросло и количество конкурентов на рынке. Сбор и анализ различных показателей приложений позволяет изменять и развивать приложение в правильном направлении при правильном анализе этих данных. Зная, как пользователь реагирует на то или иное изменение, как пользователь ведет себя в приложении, проще изменять приложение, делая его максимально удобным и привлекательным. Эти изменения позволяют разработчикам повышать прибыль с каждого пользователя. Для мобильных игр большую часть прибыли приносят «киты»: пользователи, которые продолжают играть в игру и вкладывают огромное количество денег по сравнению с другими пользователями. Статистика может помочь выявить, как

данная группа пользователей пользуется игрой. Подробная информация об игроке помогает понять, как можно удержать его в игре как можно дольше. С помощью статистики можно выявить, когда игрок хочет покинуть игру. Данная информация позволяет успеть среагировать и сделать что-нибудь индивидуальное для этого игрока для его удержания.

Когда компания понимает, что для дальнейшего улучшения своих продуктов, им нужно больше качественных данных, эта компания ищет средства реализации данной потребности. Подбор или реализация необходимых инструментов может занять длительное время. Для того чтобы посмотреть на практике выгоду от использования систем сборов статистики, удобно предварительно посмотреть ее возможности с помощью легких и очень гибких инструментов, которые настраиваются в кратчайшие сроки.

В последнее время все больше и больше набирает популярность способ предоставления услуг SaaS (*software as a service*) [20] из-за удобства использования услугами, которые предоставляются. Все вычисление, управление и данные находятся в облаке, а клиент взаимодействует с этим только с помощью веб-приложения либо посредством API [17]. Данный способ использования использовался в разрабатываемом модуле статистики.

С помощью удобного просмотра статистических данных, разработчик может понять, как стоит улучшить свое приложение, чтобы сделать его большее привлекательным для клиентов. Такая статистика может в определенный момент времени показать, что не так с приложением и позволить вовремя среагировать на событие. Примером может послужить статистика по ошибкам в приложении, если после обновления некоторая категория ошибок резко выросла, то стоит обратить на это внимание.

Для того чтобы попробовать все преимущества анализа статистики, разработчикам удобно было бы на первых этапах развития попробовать небольшой, легко настраиваемый модуль для проверки выгодности вложе-

ния средств в это направление. Такого рода сервисов на рынке не было выявлено.

Основное место для применения такого модуля статистики, рассчитанного на множество проектов, где нужно собирать небольшое количество информации для ощутимого улучшения продукта – «игровые джемы». Игры на таких мероприятиях разрабатываются от 1 до 30 дней и выходят на судейство в течении некоторого времени. В это время можно собирать статистику и улучшать игровой опыт для получения более качественной игры. Есть множество примеров, где игры на таких мероприятиях получались отличными, но, если у разработчиков было бы больше данных, они смогли бы сделать игру более комфортной для игроков.

Таким образом, целью научно-исследовательской работы становится разработка веб-приложения для сбора статистики по использованию мобильных игр.

Цель и задачи работы

Целью данной работы является разработка веб-приложения для сбора статистики по использованию мобильных игр. Сбор данных необходим для улучшения основных показателей игры и для повышения качества пользовательского опыта.

Для решения поставленной цели необходимо решить следующие задачи:

1) спроектировать модуль для сбора статистики;
2) разработка модуля для сбора статистики для игровых приложений;
3) разработка веб-приложения для модуля статистики;
4) проверить работу модуля на реальных приложениях.

Структура и объем работы

Работа состоит из введения, четырех разделов, заключения, библиографии и приложения. Объем работы составляет 32 страницы, объем библиографии – 20 источников.

Первый раздел описывает предметную область, в рамках которой ведется данная работа.

Во втором разделе описывается проектирование модуля сбора статистики для игровых приложений.

В третьем разделе описывается реализация модуля сбора статистики и веб-приложения.

В четвертом разделе приведена информация о результатах использования разработанного модуля в реальных приложениях.

В заключении суммируются основные результаты работы и описываются направление дальнейших исследований.

1. Обзор научных работ по тематике задания

1.1. Создание микросервисов.

Книга [3] посвящена программированию микросервисов — небольших автономных компонентов, позволяющих добиться модульности и отказоустойчивости любой программы. Книга дает необходимые знания для построения стабильной архитектуры и программирование стабильного модуля, каким и должен быть разрабатываемый модуль для сбора статистики.

Книга рассказывает о том, с какими проблемами данный тип сервисов чаще всего сталкивается, наилучшие пути решения этих проблем и как можно улучшить работу подобных сервисов в целом. Большая часть книги уделена способам коммуникации между сервисами. Вся эта информация применима в разработке своего сервиса.

Данная книга необходима для улучшения структуры сервера, с которым будет взаимодействовать разрабатываемое веб-приложение.

1.2. Mastering Web Application Development with AngularJS

Данная книга [5] является хорошим введением в разработку веб-приложений на фреймворке AngularJS. В книге приведено достаточно подробных примеров и «подводных камней», которые встречаются при разработке своего веб-приложения.

AngularJS JavaScript-фреймворк с открытым исходным кодом. Предназначен для разработки одностраничных приложений [9]. Его цель - расширение браузерных приложений на основе MVC-шаблона, а также упрощение тестирования и разработки.

В книге обширно рассмотрены варианты взаимодействия веб-приложения с сервером и представлены рекомендации по проектированию их взаимодействия.

1.3. Design Better Games! Flow, Motivation, & Fun

Данная статья [2] рассказывает и показывает на успешных примерах, как игры манипулируют состоянием человеческого потока, заставляя играть, ни на что не отвлекаясь, и постоянно возвращаться в игру. Статья

рассматривает основные психологические основы для игрока: поток для непрерывной игры, мотивация, чтобы заставить игрока играть дальше и веселье, которое игрок получает от всего процесса. Зная метрики, которые нужно поддерживать для такого успеха, можно целенаправленно их собирать. В статье приводится множество способов и успешных примеров, как достичь расположения игрока и заставить его играть в игру дольше. При повышении времени жизни игрока в игре повышается и качество собираемых данных о каждом игроке, что положительно сказывается на качестве разрабатываемого сервиса.

Обзор существующих решений

Google analytics

Одна из самых популярных и доступных систем сбора и анализа данных [4], предназначенных в первую очередь для веб-сайтов. Сервис плотно интегрирован с другими решениями компании Google. Изначально сервис собирал данные о том, откуда посетители приходят, как долго они остаются на сайте и где они находятся географически.

Эту аналитику иногда используют в игровой индустрии. Минусом данной системы является долгая настройка всей системы, сложное создание необходимых вам графиков для быстрого просмотра. Долгая интеграция и понимание, как именно данные собираются и выводятся не подходят для быстрого модуля сбора статистики.

Redash

Прекрасный инструмент для визуализации данных с любой базы данных [6]. Имеет прекрасную настройку досок, для удобного просмотра статистики.

Минусом платформы является ее цена, если пользоваться платформой как сервисом, а не разворачивать решение на своих серверах. Необходимо решать, в какой базе данных хранить ваши данные и каким способом туда эти данные записывать. Трудозатратность на настройку такого реше-

ния велико и оно не универсально для сбора статистики по мобильным играм.

Unity analytics

Удобный сервис для тех задач, которые необходимо решить в данной работе [7]. Удобная отправка данных и настройка доски с необходимой статистикой сделаны для игровых приложений (Unity – игровая платформа для создания игр).

В платформу сразу встроены все внешние метрики приложения (ARPU, Retention, LTV и др.), так и поддержку собственных событий. Сервис помогает настроить сбор данных так, чтобы лучше понимать своих игроков.

По причине того, что данную платформу можно использовать только на Unity, она не подходит для решения задач текущей работы.

2. ПРОЕКТИРОВАНИЕ

2.1. Архитектура модуля

На рисунке 1 представлена общая архитектура модуля сбора статистики и его взаимодействие с остальными компонентами.

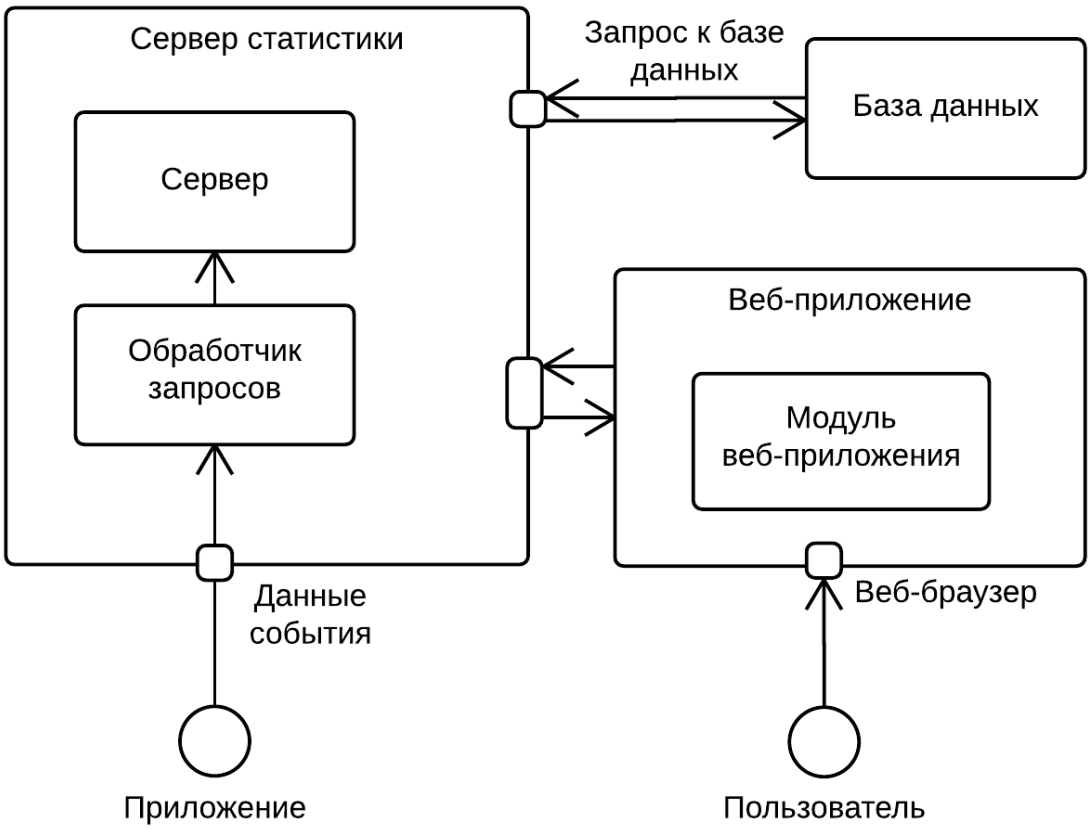


Рис. 1. Архитектура модуля сбора статистики

Архитектура модуля разделена на три уровня:

1) веб-интерфейс для работы с серверной частью, просмотра и анализа данных;

2) серверная часть, которая отвечает за сбор и анализ данных, который поступают с приложений извне;

3) набор библиотек (плагинов), которые будут подключать приложения на различных платформах для отправки данных.

Непосредственно с самим сервером статистики может взаимодействовать только приложение, которое отправляет данные о своих событиях. Пользователь может взаимодействовать с сервером только посредством

веб-приложения для просмотра данных и их анализа. Пользователь может добавлять и удалять новые приложения через веб-приложение.

Серверная часть должна обрабатывать входящие запросы на получение и добавление данных и работать с клиентами посредством RESTful [19] интерфейса. Все команды должны быть полностью документированы и доступны для разработчиков.

Все возможные действия с модулем статистики должны совершаться с помощью публичного API к серверу. Основным критерием к модулю является понятное, хорошо документированное API. Если вызов метода произошел некорректно, сервер должен выслать подробное описание ошибки и рекомендации как исправить данную ошибку, если в данных клиента присутствует ошибка. Данный подход к документированию и интеграции существенно облегчит интеграцию сервиса конечным клиентам.

В качестве серверной платформы была выбрана Node.js [9] из-за преимуществ в многопоточности и асинхронной обработки множества запросов. Запросы на сервер планируются в большом количестве, но сами запросы содержат небольшое количество данных и время обработки этих данных также невелико.

2.2. Описание приложений

Каждое приложение описано в конфигурационном файле в формате JSON. У каждого приложения в конфигурационном файле указаны все возможные события, которые оно принимает. В каждом событии указываются следующие группы полей:

- идентификация (имя события);
- исходные данные – уникальные, необходимые и опциональные поля. Данные события в строковом представлении;
- обработка данных – список статистики, которая агрегирует данные данного события.

Статистика содержит в себе следующие поля:

- имя статистики;

- описание статистики;
- тип вывода статистики (таблица или график);
- список агрегирующих функций.

Новые приложения должны добавляться с помощью API запроса к серверу с данными нового приложения и данными о сессии (авторизации пользователя). Валидация приложения происходит на стороне сервера. В случае ошибки – сервер отсылает ответом полное описание ошибки.

2.3. Взаимодействие с модулем

На рисунке 2 представлена диаграмма вариантов использования модуля сбора статистики.

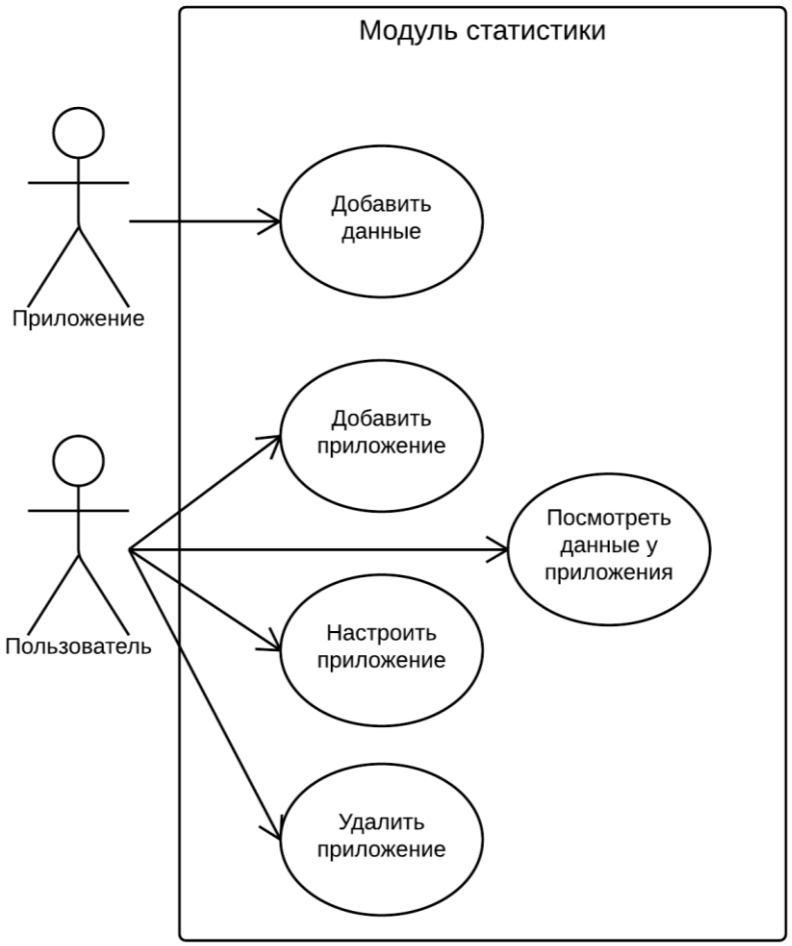


Рис. 2. Диаграмма вариантов использования

Выделено два актера, которые могут взаимодействовать с разрабатываемым модулем.

Актер «Приложение» отправляет данные на сервер. Данные передаются в текстовом формате и содержат в себе данные о каком-либо событии. Примером события может является победа на уровне. Все данные передаются с помощью HTTP-запросов.

Актер «Авторизированный пользователь» может взаимодействовать с сервером с помощью веб-приложения. Пользователь может добавлять, настраивать и удалять приложения, от которых сервер статистики может принимать данные. У каждого зарегистрированного приложения создается страница на веб-интерфейсе, где пользователь может посмотреть данные и всю статистику по ним.

Веб-интерфейс должен позволять добавлять новые приложения, добавлять к созданным приложениям события и как выводить по данным статистику. Для описания статистики необходимо выбрать какими функциями данные будут последовательно обрабатываться и в каком формате выводиться на экран. При добавлении приложения, функционал сбора данных для этого приложения должен добавляться сразу, без перезагрузки сервера.

Приоритетной библиотекой является реализация JavaScript-библиотеки, поскольку разработка большинства проектов ведется именно с помощью этой технологии. Проекты, на которых будет проводиться тестирование, разработаны на JavaScript.

2.4. Обработка данных

Все данные отсылают клиенты мобильных приложений. Разработчик встраивает библиотеку и вызывает функции отправки данных в необходимых местах. Если подключение к интернету отсутствует, необходимо сохранить те данные, которые не были доставлены до сервера для дальнейшей отправки в будущем, когда будет подключение к сети интернет.

Все данные, которые отправляются на сервер модуля сбора статистики отправляются асинхронно. На рисунке 3 представлена диаграмма взаимодействий модуля сбора статистики.

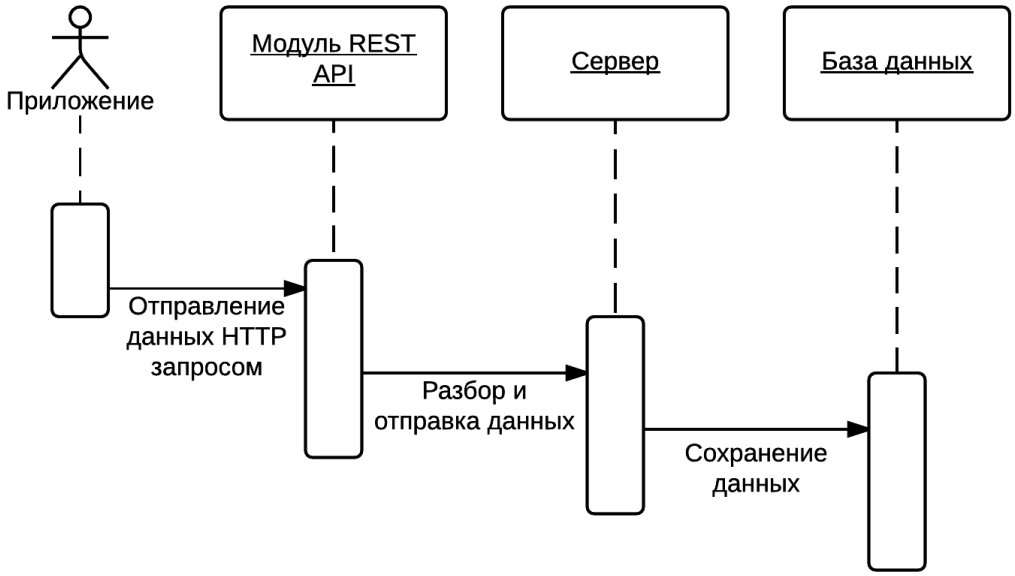


Рис. 3. Диаграмма взаимодействий

На диаграмме взаимодействия продемонстрировано, как идет поток данных через сервер от приложений. Все модули действуют асинхронно. Модуль REST API и сервер модуля сбора статистики могут быть масштабированы горизонтально путем увеличения количества узлов.

3. РЕАЛИЗАЦИЯ

Серверная часть модуля реализована на платформе Node.js. модуль обрабатывает все приходящие данные с приложений и распределяет их по базам данных, которые данным модулем и анализируются. База данных не привязана к разработанному модулю и может быть легко заменена любой другой путем подмены реализации адаптера базы данных.

В качестве серверного веб-фреймворка был выбран ExpressJS [8]. Этот простой, но мощный фреймворк обеспечивает достаточный контроль над всем приложением и имеет простой порог вхождения для начала разработки своего приложения. ExpressJS позволяет быстро разработать необходимый вам функционал, удобен для реализации REST сервисов из-за мощной системы «роутинга», привязки по каждому HTTP адресу своего обработчика.

Приложение оперирует данными в формате JSON с заранее неизвестной структурой. По этой причине в качестве базы данных выбрана MongoDB [11], одна из самых популярных объектно-ориентированных баз данных. Плюсом MongoDB является так же то, что она имеет отличный баланс скорости и стойкости, отлично подходящей для задачи сбора статистики. NoSQL база данных была взята по той причине, что заранее неизвестна схема таблиц, которые используются в приложениях. NoSQL с возможность сохранять любые данные подходит для решения текущей задачи.

Архитектура приложения такова, что имеется общий интерфейс к модулю базы данных и на данный момент реализовано два адаптера к базам данных: MongoDB и Couchbase. При необходимости развернуть модуль на другой базе данных, необходимо реализовать интерфейс адаптера и добавить его в список поддерживаемых баз данных. Адаптер содержит в себе методы подключения и отключения к базе данных, сохранения и загрузки данных.

Модули предметной области

На рисунке 4 представлены модули предметной области.

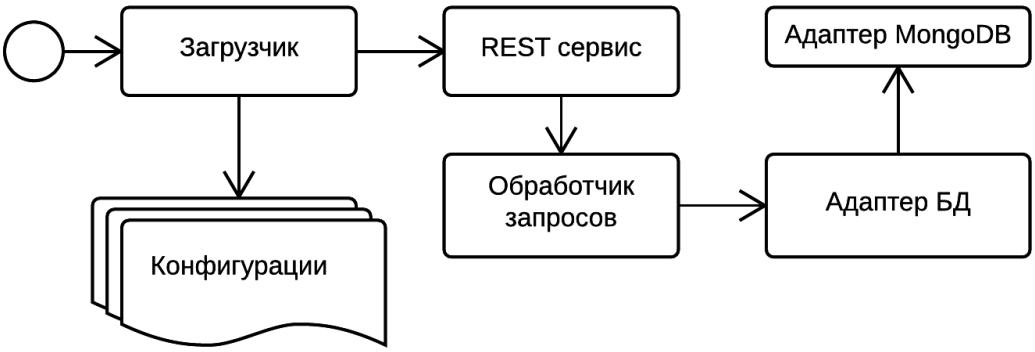


Рис. 4. Модули предметной области

Ниже приведено краткое описание каждого модуля приложения.

Модуль **Загрузчик** необходим для инициализации всего модуля и загрузки всех пользовательских приложений (конфигураций). Загрузчик запускает работу ExpressJS и инициализирует модуль REST сервиса.

Модуль **REST сервис** отвечает за то, чтобы для каждого приложения, был указан его HTTP адрес к событиям приложения и по этим адресам можно было делать HTTP запросы для отправки данных. Для каждого адреса указан свой обработчик из модуля обработчика запросов.

Модуль **Обработчик** запросов работает с каждым запросом, который направлен по адресу приложения. Модуль обращается к адаптеру базы данных для получения информаций, занимается обработкой данных при сохранении или получении данных и в случае некорректных данных отправляет код ошибки обратно пользователю.

Модуль **Адаптер БД** необходим для стандартизации работы с базами данных. Модуль обладает интерфейсом для подключения, загрузки и сохранении данных и отключении от базы данных. На рисунке 4 показано, что в текущей версии используется адаптер MongoDB для работы с базой данных MongoDB.

Описание приложений

При добавлении нового приложения, на сервер добавляется новый конфигурационный файл с описанием нового приложения. При загрузке

сервера или по событию изменения приложения, этот файл считывается заново и добавляет пути в REST-сервис для сохранения данных этого приложения. Динамическое отслеживание изменений в конфигурациях позволяет пользователям добавлять приложения во время работы модуля. На рисунке 5 приведен пример конфигурационного файла.

```
"app_name": "BeatsofNightmare",
"app_key": "bon",
"app_is_show": true,

"app_events": [
{
      "event_name": "highscore",
      "event_fields_unique" : ["username"],
      "event_fields_required" : ["score"],
      "event_fields_optional" : [],
      "event_stats": [
      {
            "stat_name": "highscore",
            "stat_snippet": "Get first 10 highscores",
            "stat_type": "table",
            "stat_fun": "Sort(score)=>Limit(10)"
      },
      {
            "stat_name": "best",
            "stat_snippet": "Get the best score",
            "stat_type": "table",
            "stat_fun": "Min(score)"
      }
      ]
}
]
```

Рис. 5. Пример конфигурационного файла

При совершении запроса, к данным добавляется hash-сумма всех отправляемых данных вместе с паролем приложения. Такая же процедура производится на стороне сервера. Если ключи совпадают – данные подтверждаются и добавляются в базу данных. Данная мера сделана для большей безопасности и снижения рисков от чужеродных запросов с целью подделать данные.

Поскольку MongoDB не поддерживает SQL запросы, а использует для получения данных из базы функцию *find()*, реализован метод, с помощью которого необходимые данные забираются для статистики.

В конфигурационном файле описывается список функций или условий, которые выполняются перед тем, как вернуть результат. Ниже представлен список из нескольких таких функций.

- Limit(N) – ограничивает выбор данных N элементами;
- Sort(key), SortReversed(key) – сортирует данные по ключю key;
- Min(), Max() – выбирает первый или последний элемент из набора данных;
- Count() – возвращает количество элементов;
- Last([hour, day, week, month, year]) – выбирает данные за указанное время или группирует данные по временному промежутку. Зависит от способа отображения статистики.

Рассмотрим следующую функцию:

```
"stat_fun": "Last(week)=>Sort(score)=>Limit(10)"
```

Данная функция выбирает данные за последнюю неделю и отдает десять лучших результатов. Данные поэтапно проходят через вызов каждой функции слева направо, перед тем, как передаются на вход для отображения графиков.

При описании конфигурации приложения, можно указать какие поля содержаться в данных этого приложения. Есть три типа полей:

- *уникальные поля* – если в данных встречается такой же набор уникальных полей, данные перезаписываются новыми данными;
- *обязательные поля* – поля, которые необходимо передавать при отправки данных на это событие;
- *необязательные поля* – поля, которые могут быть незаполненными при отправки данных на это событие.

Веб-интерфейс

В качестве веб-интерфейса решено сделать одностраничное веб-приложение, которое позволит пользователям удобно и быстро работать с данным модулем статистики. В качестве фреймворка был выбран AngularJS [9].

В процессе работы AngularJS показал себя как простой и быстрый в использовании инструмент, позволяющий за короткое время реализовать приложение и развернуть его на сервере.

При обращении к каким-либо данным, осуществляет асинхронный запрос к данным на сервере. После того, как сервер ответит и вернет запрашиваемые данные в формате JSON, AngularJS заполнит данные на странице по необходимому шаблону. Созданы следующие шаблоны: главная страница, список приложений, список событий, шаблон события и шаблон показа статистики. Шаблон представляет собой HTML документ с указанными в определенных местах специальными метками, в которые отображаются данные, связанные с приложением.

В качестве библиотеки для отображения графиков была выбрана Google Charts [10], одна из самых популярных и простых библиотек в своей категории. Она позволяет быстро и просто рисовать различные варианты графиков. Для модуля статистики основным способов вывод данных является график с линиями, пример приведен на рисунке 6. На каждую линию можно навести курсором мыши для всплывавших подсказок (значение в определённое время, вывести полное название линии).

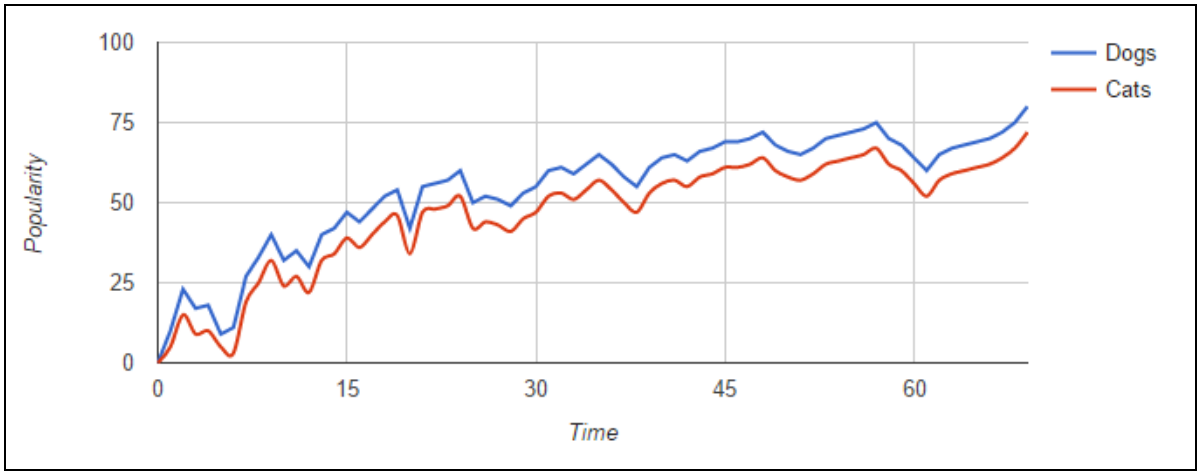


Рис. 6. Пример графика

Другим способом вывода информации является таблица. Способ вывода данных указывается в настройках события приложения. Пример таблицы приведен на рис. 7. Каждое поле может быть целым числом, строкой

или флагом. Каждый столбец можно сортировать для удобного просмотра, кликнув курсором мыши на заголовок. Названия заголовков берутся из конфигурационного файла.

	Nickname	Score	Is Hardcore
1	Insality	10,000	✓
2	Joejah	12,500	✓
3	Glory	7,000	✓
4	Pepsikat	8,000	✗

Рис. 7. Пример таблицы

Веб-интерфейс модуля статистики позволяет просматривать все события у каждого приложения, каждую статистику, описанную в конфигурационном файле приложения в виде графиков или таблиц.

На данный момент, список приложений и его событий выводятся простым списком. На рисунке 8 приведен пример.

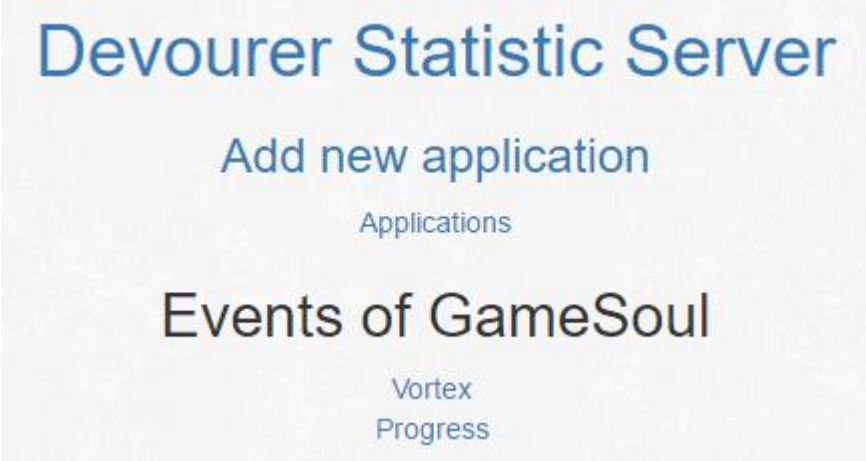


Рис. 8. Пример выбора события приложения GameSoul

Добавление нового приложения происходит через личный кабинет пользователя. Пользователь заполняет форму и с помощью API сервера модуля статистики, данные отправляются на сервер и добавляются в общий список приложений. Авторизация на данный момент происходит через социальную сеть. Сделано это из-за простоты и скорости в реализации. В дальнейшем планируется добавить возможность собственной регистра-

ции пользователей или вход через другие популярные социальные сети разработчиков (GitHub, Google, Facebook и др.).

На рисунке 9 приведен экран временной формы создания нового приложения. События и статистику можно добавлять в неограниченном количестве. Редактирование приложения происходит на этой же форме.

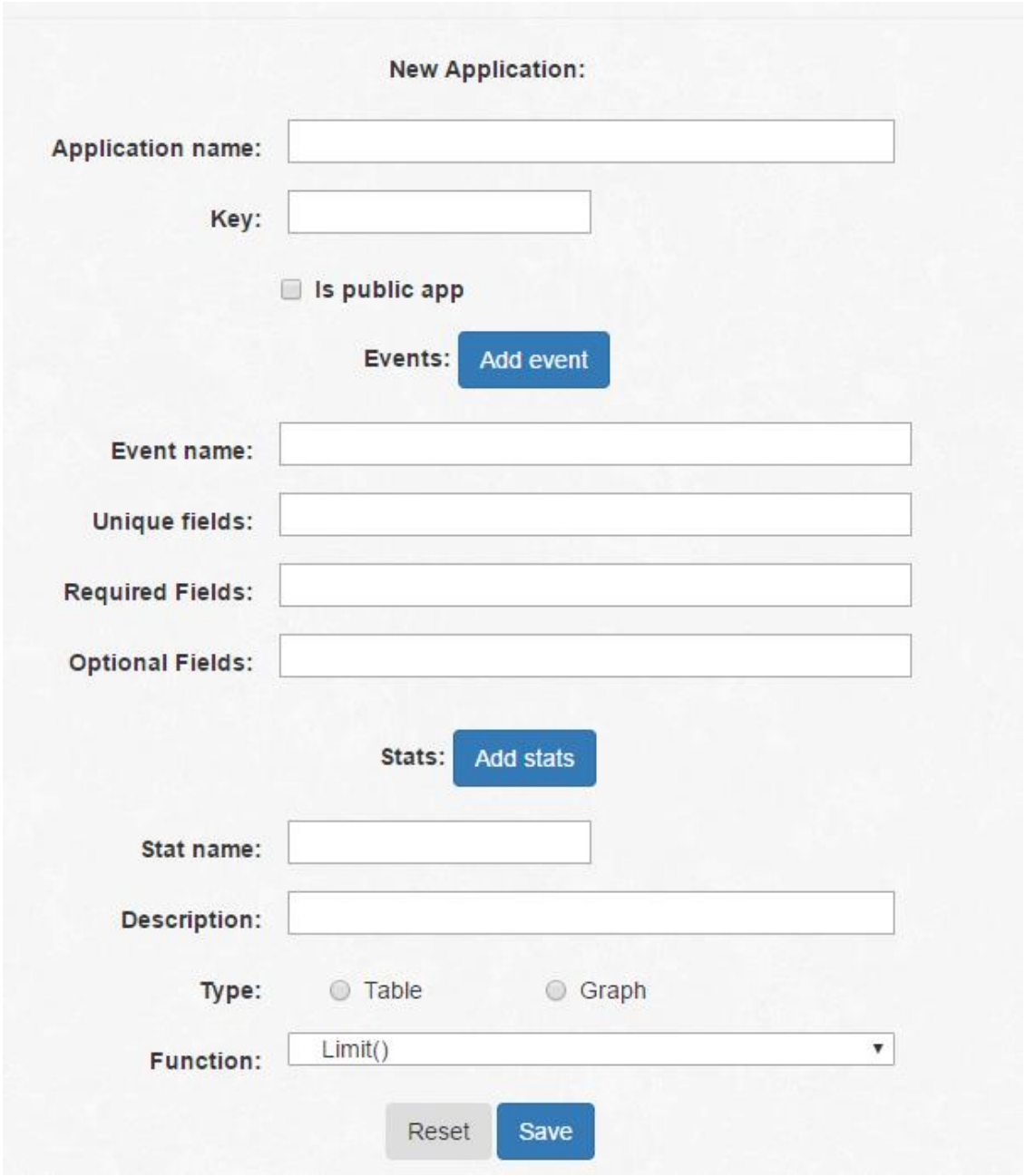


Рис. 9. Форма создания приложения

Веб-приложение является визуальным взаимодействием с API сервером и позволяет добавлять и редактировать приложения, а также просматривать статистику по своим приложениям.

4. ТЕСТИРОВАНИЕ

В данной главе будет рассмотрен опыт эксплуатации модуля сбора статистики на примере двух игр. Первое использование модуля на практике, в производственной эксплуатации, производилось именно на этих проектах.

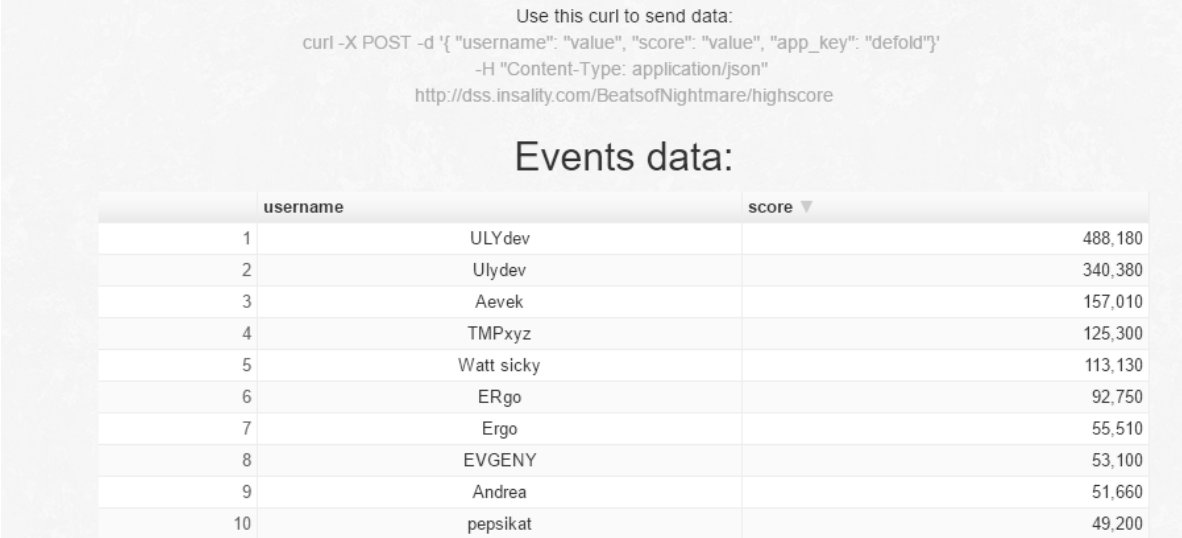
Модуль разрабатывался с упором на мгновенную интеграцию и оценку того, насколько нужно собирать полную статистику вашего приложения. Два проекта из ниже осмотренных – проекты с «игровых джемов», где игры разрабатываются в короткий срок (2-3 дня). Небольшая статистика позволяет выявить критичные моменты в игре и получением игрового опыта пользователями. Подробно моменты рассмотрены ниже.

4.1. Beats of Nightmare

Данная игра разрабатывалась для «Ludum Dare 33», мероприятие, где нужно создать игру за 48 часов. Игра включает в себя соревновательный режим по игровым очкам, которые игрок получает за прохождение одного игрового сеанса. На рисунке 10 показан логотип игры.

Рис. 10. Логотип «Beats of Nightmare»

Лучшие результаты выводятся всем остальным игрокам. Необходимо было собирать информацию, на каком этапе игры игроки проигрывали. Цель была улучшить игру таким образом, чтобы игроки увидели как можно больше игрового контента. Первая версия модуля была написана именно на этом мероприятии. Он включал в себя простое отображение данных и сохранение их с клиентов игр. На рисунке 11 приведен пример лучших результатов по окончании соревнования в игре.



	username	score
1	ULYdev	488,180
2	Ulydev	340,380
3	Aevek	157,010
4	TMPxyz	125,300
5	Watt sicky	113,130
6	ERgo	92,750
7	Ergo	55,510
8	EVGENY	53,100
9	Andrea	51,660
10	pepsikat	49,200

Рис. 11. Таблица лучших игроков

Удобным моментом оказалось то, что данные из события можно получать внутри игры с помощью HTTP GET запроса, что позволило выводить имена лучших игроков прямо в игре. Неудобным моментом с созданием таблицы лучших игроков, что имена были уникальные и счет необходимо было переписывать, если только он лучше, чем предыдущий. Для решения данной проблемы необходимо добавить правила перезаписывания по уникальным полям события. Такими правилами могут выступать «больше-лучше» или «меньше-лучше».

В процессе голосования собиралась статистика по прохождениям игровых этапов пользователями. В первый день игроки до второго этапа доходило 15 % игроков, до 3 этапа около 5 %. Зная, что у игроков есть проблемы со сложностью игры, были предприняты меры по облегчению пер-

вых этапов, чтобы игроки смогли больше поиграть в игру и собрать больше положительных отзывов. К концу голосования количество людей, прошедших первый уровень увеличилось до 60 %, а второй до 40 %.

4.2. Ancient Invaders

Данная игра разрабатывалась для GamesJamKanobu в 2016 году [15]. Разработка длилась 30 дней и участвовала в соревновании с другими играми за призовые места на международной конференции WhiteNights’2016 Moscow [13]. На рисунке 12 приведен игровой снимок игры Ancient Invaders.

Рис. 12. Игровой снимок Ancient Invaders

Цель внедрения статистики была схожа с прошлым экспериментов в Beats of Nightmare. Собирались данные о том, насколько игроки далеко проходят, за сколько минут они проходят игру, за сколько секунд они добирались до каждой из игровых целей и лучший счет игроков.

Данные по скорости прохождения были необходимы для того, чтобы настроить игру для лучшего пользовательского опыта, которая игра может

дать, будучи сделанной на игровом хакатоне. Необходимо было сделать игровое время в игре для большинства игроков около 4 минут, не больше. Игра показывалась на конференции, где много игроков периодически подходит и смотрит игру.

В первую неделю среднее время прохождения игры было около 8-10 минут. Без подобной статистики узнать о таком было бы просто невозможно. На рисунке 13 приведен пример одной из собираемых метрик по игре.

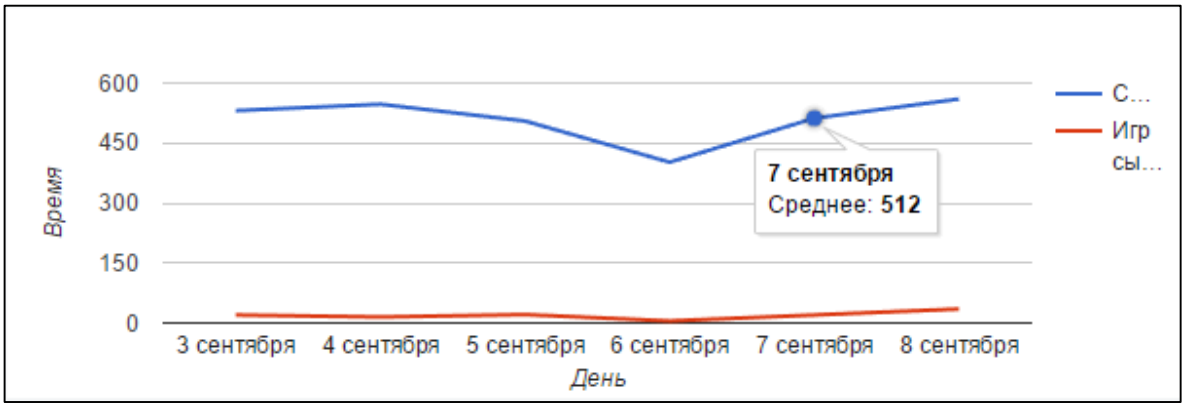


Рис. 13. Статистика по Ancient Invaders

Путем улучшения обучения в игре и облегчению некоторых игровых процессов, был достигнут желаемый результат по сокращению игровому времени, что положительно сказалось на восприятии игры игроками.

При подсчете количества игроков, которые достигли определенного этапа в игре, не хватало поля счетчика, который бы инкрементировался каждый раз, когда событие приходит на сервер. Обходным вариантом является суммирование количества всех событий, но счетчик был бы более логичным в этом случае.

4.3. Морской бой

Морской бой [16] – игра, разрабатываемая под новую платформу «Одноклассников» «Игры в сообщениях». Игра представляет собой стандартную всем известную игру морской бой, где два игрока в реальном времени соревнуются между собой. В игре добавлены различные бонусы

или усиления, которые игрок может применять против другого игрока для более интересного игрового опыта. Игры в сообщениях представляют такой тип игр, где в социальных сетях, общаясь с другими людьми, можно непосредственно в сообщениях начать игру с собеседником. На рисунке 14 представлен игровой снимок из игры «Морской Бой».

Рис. 14. Игра «Морской бой» в «Одноклассниках»

Разрабатываемый модуль статистики использовался при прототипировании игры и понимания, насколько игра соответствует ожиданиям разработчиков и игроков. На этапе прототипирования регистрировались и анализировались следующие метрики:

- на каком этапе заканчивали играть новые игроки;
- как много игр было начато и завершено;
- где пользователи покупают игровую валюту и сколько;
- длина игровой сессии.

Данные метрики помогли улучшить основные показатели игры (ARPU, Retention, LTV) в несколько раз. Когда игра выпускается как про-

тотип на небольшую аудиторию, очень важно подробно анализировать, как игроки играют в игру и понять, в какую сторону необходимо развивать приложение. На рисунке 15 приведена воронка новых пользователей.

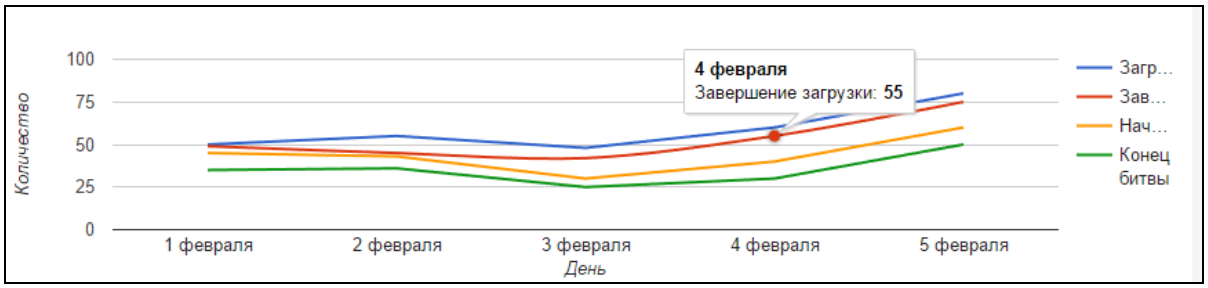


Рис. 15. Воронка новых пользователей в «Морском бою»

Неудобным моментом оказалось наблюдение большого количества графиков за раз, необходимо добавить доски, на которые можно добавлять несколько статистик из разных событий, чтобы просматривать их на одной странице. Сами графики получаются довольно информативными и помогают разработчикам принимать решения по развитию приложения.

После утверждения жизнеспособности продукта и переход разработки в основную стадию, вся система аналитики была переведена на внутреннюю аналитику компании. В большом производственном решении небольшой модуль сбора статистики не справится с миллионным наплывом пользователей.

ЗАКЛЮЧЕНИЕ

Данная работа посвящена разработке веб-приложения для сбора статистики по использованию мобильных игр. Разработанное веб-приложение позволяет в краткие сроки добавить сбор статистики в игру. Статистика о том, как игроки играют в игру, позволяет легче улучшать продукт с учетом реального игрового опыта игроков.

Основные результаты работы

В ходе выполнения работы были получены следующие основные результаты:

1) спроектирован модуль для сбора статистики для игровых приложений;

2) разработан модуль для сбора статистики для игровых приложений;

3) разработано веб-приложение для модуля статистики;

4) протестирована работа модуля на реальных приложениях.

Основные направления дальнейшей работы

Работа по представленной теме может быть продолжена в следующих направлениях:

1) улучшение удобства веб-страницы приложения;

2) добавление внешней игровой статистики для приложений;

3) улучшение качества сервиса и создание платных тарифов.

ЛИТЕРАТУРА

1. Brown E. Web development with Node and Express. – USA: O'Reilly Media, 2014. – 332 p.

2. Murphy C.A., Chertoff D.D., Guerrero M.A. Design better games! Flow, Motivation, & Fun. // Ted conference. 2010. – 37 p.

3. Newman S. Building Microservices. – USA: O'Reilly Media, 2015. – 280 p.

4. Google Analytics. [Электронный ресурс] URL: https://www.google.com/analytics/ (дата обращения: 12.06.2017).

5. Kozlowski P.A., Bacon P.D. Mastering Web Application Development with AngularJS. – Packt Publishing, 2013. – 372p.

6. Redash. [Электронный ресурс] URL: https://redash.io (дата обращения: 12.06.2017).

7. Unity Analytics. [Электронный ресурс] URL: https://unity3d.com/ru/services/analytics (дата обращения: 12.06.2017).

8. Документация NodeJS библиотеки «ExpressJS». [Электронный ресурс] URL: http://expressjs.com/ (дата обращения: 10.01.2017).

9. Документация по технологии AngularJS. [Электронный ресурс] URL: https://angularjs.org/ (дата обращения: 14.02.2017).

10. Документация по технологии Google Charts. [Электронный ресурс] URL: https://developers.google.com/chart/ (дата обращения: 12.02.2017).

11. Документация по базе данных MongoDB. [Электронный ресурс] URL: https://docs.mongodb.com/ (дата обращения: 10.02.2017).

12. Документация по технологии NodeJS. [Электронный ресурс] URL: https://nodejs.org/en/ (дата обращения: 10.02.2017).

13. Конференция White Nights. [Электронный ресурс] URL: http://wnconf.com/ (дата обращения: 05.05.2017).

14. Статистика мобильного рынка. [Электронный ресурс] URL: http://www.gamasutra.com/blogs/JukkaHilvonen/20160405/269664/Game_design_lenses_help_target_your_games.php (дата обращения: 10.06.2016).

15. Проект Ancient Invaders на GamesJam. [Электронный ресурс] URL: http://gamesjam.org/2930/ (дата обращения: 25.06.2017).

16. Проект «Морской Бой» на одноклассниках. [Электронный ресурс] URL: https://ok.ru/game/piratesbattles (дата обращения: 02.03.2017).

17. Описание API технологии. [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/API (дата обращения: 12.03.2017).

18. Описание NoSQL технологии. [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/NoSQL (дата обращения: 20.04.2017).

19. Описание REST технологии. [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/REST (дата обращения: 13.03.2017).

20. Описание SaaS технологии. [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/SaaS (дата обращения: 13.03.2017).