

Федеральное государственное автономное  
образовательное учреждение  
высшего образования  
**«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Институт Космических и Информационных Технологий  
институт  
Информационные системы  
кафедра

УТВЕРЖДАЮ  
Заведующий кафедрой ИС  
\_\_\_\_\_ Л.С. Троценко  
подпись инициалы, фамилия  
«13» июня 2018 г.

## **БАКАЛАВРСКАЯ РАБОТА**

09.03.02 Информационные системы и технологии

Минимизация инвестиционных затрат на внутриигровые предметы  
средствами веб-портала

Руководитель	_____	<u>С.А. Виденин</u>
	подпись, дата	инициалы, фамилия
Выпускник	_____	<u>М.А. Бобин</u>
	подпись, дата	инициалы, фамилия
Нормоконтролер	_____	<u>Ю.В. Шмагрис</u>
	подпись, дата	инициалы, фамилия

Красноярск 2018

Федеральное государственное автономное  
образовательное учреждение  
высшего образования  
**«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Институт Космических и Информационных Технологий  
институт  
Информационные системы  
кафедра

УТВЕРЖДАЮ  
Заведующий кафедрой ИС  
\_\_\_\_\_ С.А. Виденин  
подпись инициалы, фамилия  
« 05 » 03 2018г.

**ЗАДАНИЕ**  
**НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ**  
**в форме бакалаврской работы**

Студенту Бобину Максиму Алексеевичу

фамилия, имя, отчество

Группа КИ14-13Б Направление 09.03.02

номер

код

Информационные системы и технологии

наименование

Тема выпускной квалификационной работы: Минимизация инвестиционных затрат на внутриигровые предметы средствами веб-портала

Утверждена приказом по университету № 4896/с от 05.04.2018г.

Руководитель ВКР С. А. Виденин, кандат пед. наук, доцент, заведующий кафедрой «Информационные системы» ИКИТ

инициалы, фамилия, должность, ученое звание и место работы

Исходные данные для ВКР: Требования к разрабатываемому сервису, рекомендации руководителя

Перечень разделов ВКР: Введение, обзор и анализ предметной области, проектирование веб-портала, разработка веб-портала, апробация веб-портала (на примере действий пользователей), заключение, список использованных источников.

Перечень графического материала: Презентация, выполненная в Microsoft Office PowerPoint 2013

Руководитель ВКР

\_\_\_\_\_

С. А. Виденин

подпись

инициалы и фамилия

Задание принял к исполнению

\_\_\_\_\_

М. А. Бобин

подпись

инициалы и фамилия

« 05 » 03 2018г.

## **РЕФЕРАТ**

Выпускная квалификационная работа по теме «Минимизация инвестиционных затрат на внутриигровые предметы средствами веб-портала» содержит 47 страниц текстового документа, 35 иллюстраций, 25 использованных источников.

**ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ВЕБ-ПОРТАЛ, LARAVEL, ЯЗЫК PHP, ЯЗЫК HTML, ЯЗЫК JAVASCRIPT, NODE\_JS, BOOTSTRAP, БАЗА ДАННЫХ, MySQL, CONTROLLER.**

Целью работы — минимизировать инвестиционные затраты на приобретение внутриигровых предметов средствами веб-портала. Для достижения поставленной цели были определены следующие задачи:

- изучить предметную область;
- рассмотреть существующие аналоги;
- разработать веб-портал.

В результате выполнения выпускной квалификационной работы был создан веб-портал, позволяющий людям минимизировать инвестиционные затраты на приобретение внутриигровых предметов.

## СОДЕРЖАНИЕ

Введение.....	4
1 Обзор и анализ предметной области .....	5
1.1    Анализ деятельности информационной системы.....	5
1.2    Обзор существующих аналогичных веб-сервисов .....	6
2 Проектирование веб-портала .....	14
2.1    SADT-диаграмма.....	14
2.2    Проектирование базы данных .....	20
2.3    Обзор выбранной архитектуры и инструментов для реализации веб-портала .....	22
3 Разработка веб-сервиса.....	30
4 Апробация веб-портала (на примере действий пользователей).....	41
Заключение .....	45
Список использованных источников .....	46

## ВВЕДЕНИЕ

На сегодняшний день многопользовательская онлайн игра Counter Strike: Global Offensive невероятно популярна, однако, легко забыть, что игра не была мгновенным лидером продаж для Valve. Изначально это была всего лишь одна из разновидностей устаревшей Counter Strike, так же в то время на рынке было предостаточно шутеров на любой вкус и цвет.

Однако, все изменилось в один миг, когда Valve представила нечто новое — декоративное виртуальное оружие, известное как «скины», что можно получать случайно в игре или купить в онлайн-магазине.

В результате, число игроков в CS:GO за 2 года выросло на 1500%.

На сегодняшний день общее число игроков составляет около 35 миллионов, а количество в онлайн за две недели около 10 миллионов пользователей.

Игроки ежедневно совершают покупки внутриигровых предметов, которые порой стоят дороже самой игры. Так, например, стоимость одного из самых дорогих предметов составляет примерно 100000\$.

В итоге, за последние годы Valve смогла создать для своих игр одну из самых мощных виртуальных экономик в истории.

Таким образом для большинства игроков актуально получить скины за минимальную цену. Наша АИС даст шанс улучшить свой инвентарь, обрести новые вещи из популярной игры при минимальной сумме депозита, который формируется исходя из оценки уже имеющихся у игрока предметов.

Основываясь на актуальности была поставлена цель ВКР: минимизировать инвестиционные затраты на приобретение внутриигровых предметов средствами веб-портала.

Для достижения поставленной цели были определены следующие задачи:

- изучить предметную область;
- рассмотреть существующие аналоги;
- разработать веб-портал.

# **1 Обзор и анализ предметной области**

Перед тем как начать проектирование информационной системы, необходимо изучить основные определения и проанализировать существующие решения на рынке для выявления особенностей и недочетов в использовании подобных систем.

## **1.1 Анализ деятельности информационной системы**

Определение информационной системы имеет сотни интерпретаций, непосредственно связанных со сферой применения той или иной системы. Федеральный закон Российской Федерации «Об информации, информационных технологиях и о защите информации» трактует понятие «информационной системы» как совокупность содержащейся в базах данных информации и обеспечивающих её обработку информационных технологий и технических средств. Российский стандарт ГОСТ Р В 51987 подразумевает под ИС «автоматизированную систему, результатом функционирования которой является представление выходной информации для последующего использования».

На сегодняшний день существует множество средств реализации информационных систем. В зависимости от того, в какой сфере планируется использование информационной системы, а также для решения какого бизнес-процесса она предназначена, в каждом средстве можно выявить свои преимущества и недостатки.

Учитывая выбранную сферу деятельности, веб-технологии являются самым оптимальным средством реализации информационной системы.

## 1.2 Обзор существующих аналогичных веб-сервисов

Данные веб-сервисы очень популярны, в рамках данной работы было решено рассмотреть наиболее известные, например, сайт «CSGOGOLD.RU»

Данный сайт имеет удобный и красивый интерфейс, показан на рисунке 1.

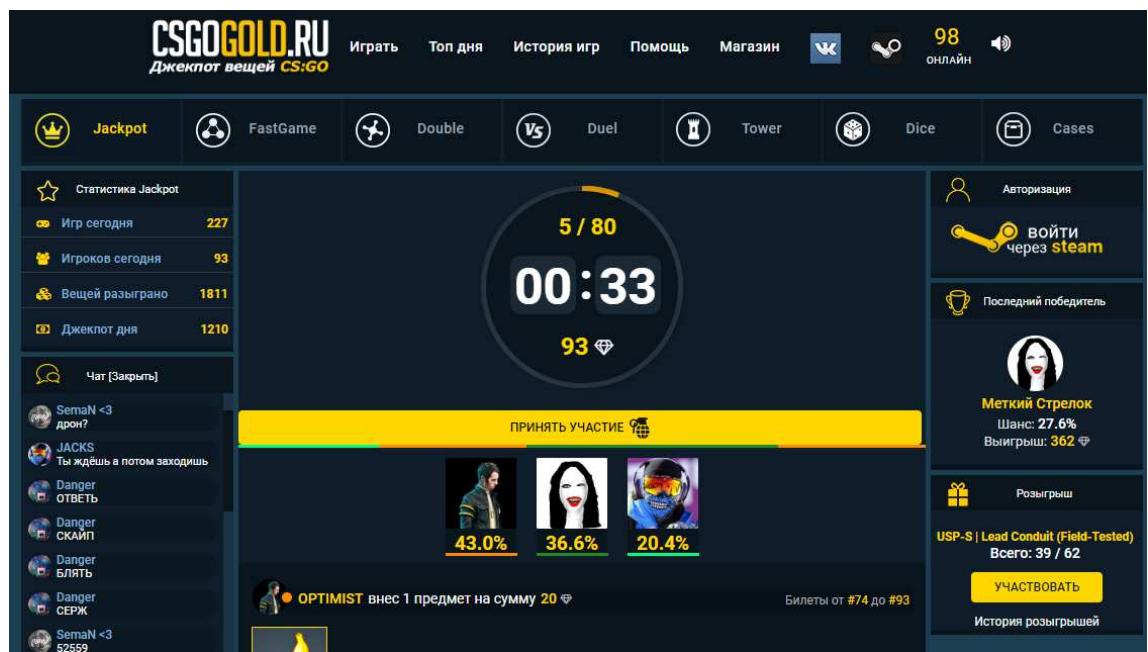


Рисунок 1 — Главная страница

Депозит вносится непосредственно скинами через steam. Они предлагают одинаковое значение поинтов для предметов независимо от того, вносите ли вы или выводите их. Пользователь всегда получает поинты, основанные на стоимости предмета, который вкладывает.

Запросы обрабатываются довольно быстро, и в истории службы не было отложенного депозита или запроса на вывод. Также на данном сайте находится магазин, в котором пользователь может приобрести скины за имеющееся у него поинты. В инвентаре представлен широкий выбор предметов по цене гораздо ниже чем на торговой площадке steam.

Однако было отмечено два существенных минуса. Первый заключается в том, что перед снятием вы должны сделать bet на семьдесят пять процентов от вашего депозита, но по заявлениям разработчиков это делается для того, чтобы трейдеры не злоупотребляли ботами. Второй минус заключается в самом алгоритме, который используется на данном сайте, а именно, вероятность выигрыша полностью зависит от количества поставленных поинтов, чем больше поинтов, тем соответственно больше шанс на победу. Данная система проиллюстрирована ниже.

На рисунке 2 user 1 внес свой предмет на сумму 20 поинтов и вероятность его выигрыша составляет 100%, т.к в игре он один.

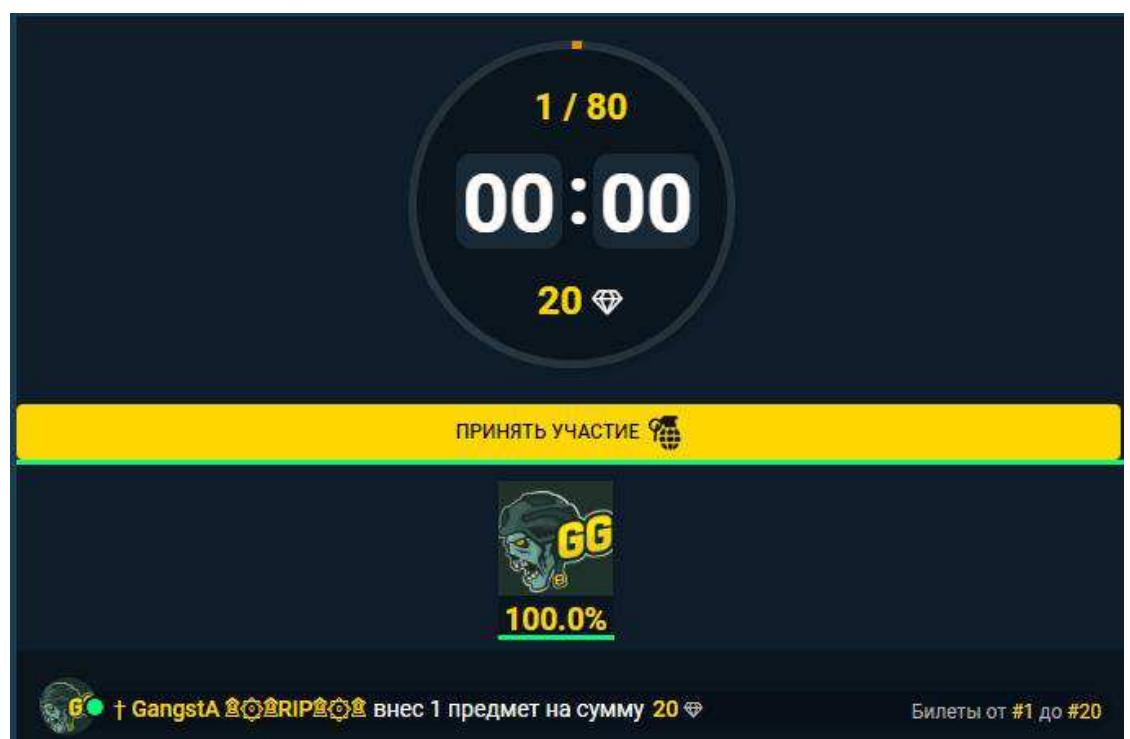


Рисунок 2 — Демонстрация розыгрыша

На рисунке 3, user 2 внес предмет на сумму 16 поинтов и шанс выигрыша user 1 сразу понизился, но все еще был выше, чем у user 2.

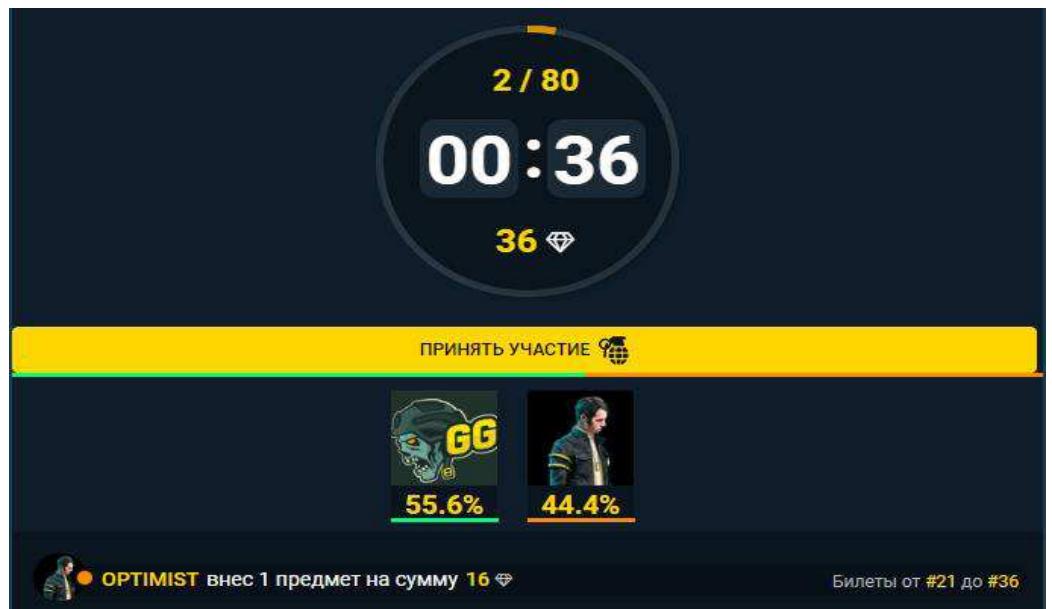


Рисунок 3 — Демонстрация розыгрыша

На рисунке 4, user 3 внес предмет на сумму 10 поинтов, т.к это оказалось меньше всего, то и вероятность выиграть у данного пользователя меньше, чем у других.

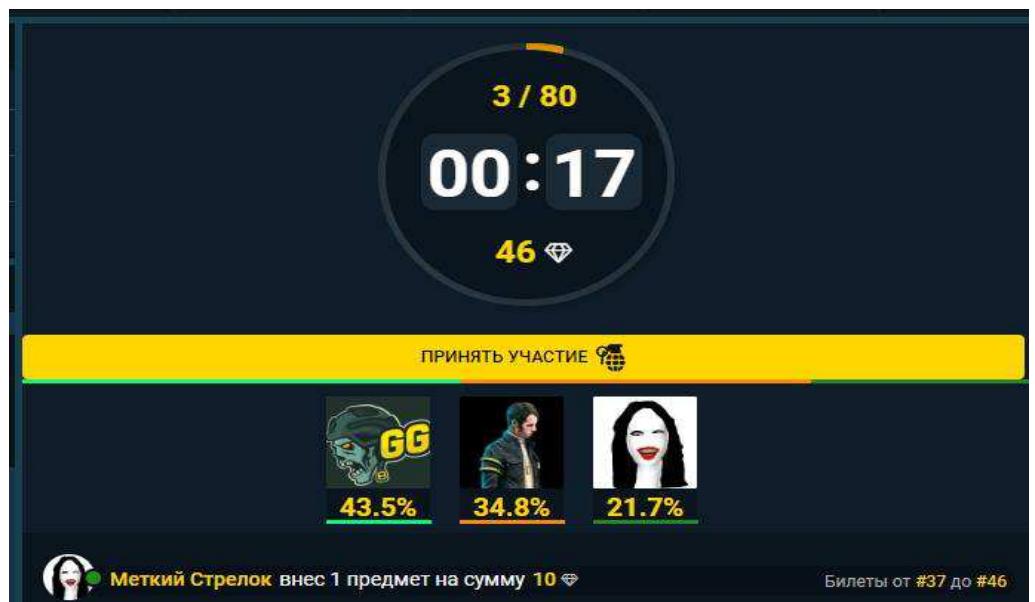


Рисунок 4 — Демонстрация розыгрыша

Так же любой из участников может увеличить свой шанс на победу, добавив еще предметы, что и сделал в свою очередь user 2, на рисунке 5, что сделало его вероятность на выигрыш выше чем у остальных.

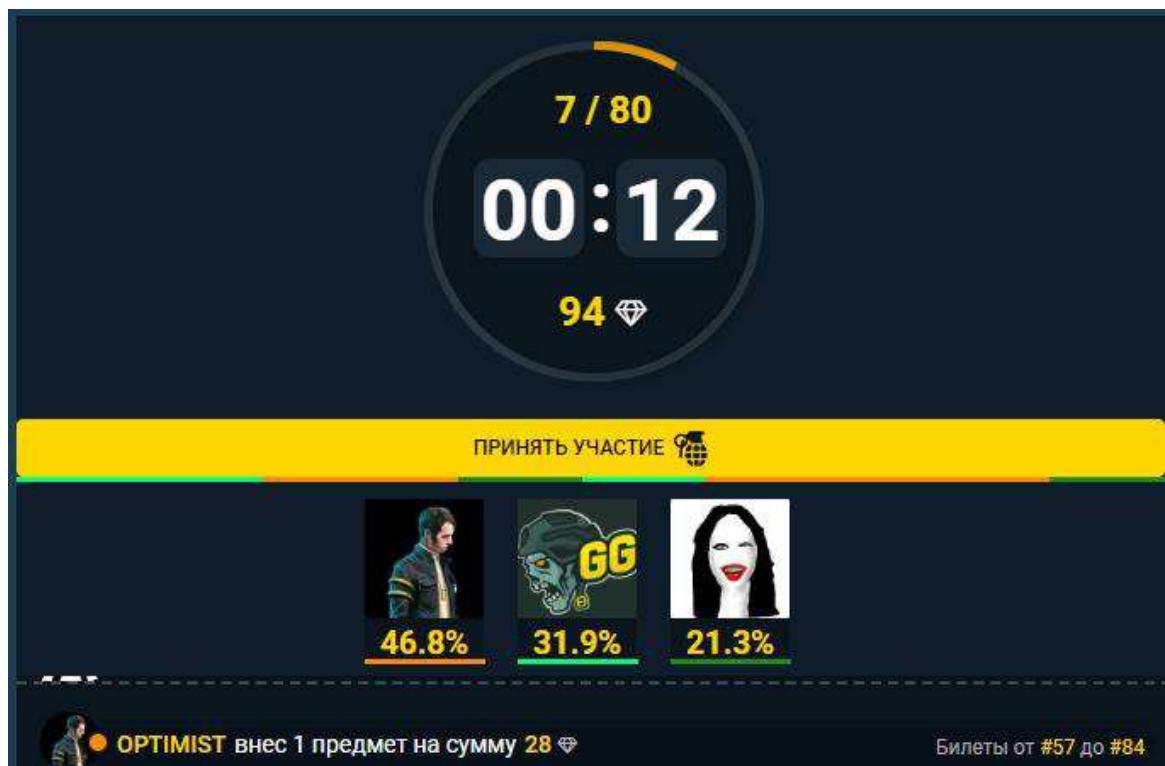


Рисунок 5 — Демонстрация розыгрыша

Все манипуляции по повышению вероятности на победы пользователи могут совершать на протяжении 60 секунд, по истечению данного времени бот автоматически запускает ленту, на которую нанесены аватары участников. Бот имеет приоритет остановить ленту на иконке пользователя, чья вероятность выигрыша выше, и он зачастую одерживает победу, так и в этот раз выиграл user 2, показано на рисунке 6.



Рисунок 6 — Демонстрация розыгрыша

В итоге был сделан вывод, что данный сервис не позволит пользователю, который вводит в игру предмет на меньшее количество поинтов чем кто-либо другой, выиграть. Точнее теоретически это возможно, но на практике маловероятно. Поэтому было решено спроектировать аналогичный веб-сервис, работающий по алгоритму, который позволяет уравнять вероятность на победу у каждого пользователя. А именно, в каждой игре будет всего 4 участника, которые могут ввести в игру любое количество поинтов на определенный цвет из четырех. Вероятность выигрыша в данном случае будет равна 25% у каждого пользователя, выигрыш будет составлять удвоенное количество поинтов. Благодаря этой системы любой участник сможет приумножить свой капитал.

Также построена use-case диаграмма, данная диаграмма представляет собой схему взаимодействия пользователя с системой, которая показывает связь между пользователем и различными случаями использования, в котором пользователь участвует. Диаграмма вариантов использования может идентифицировать различные типы пользователей системы и различные варианты использования и часто может сопровождаться другими типами диаграмм.

## Общие компоненты:

– Актер: представляет пользователя, организацию или внешнюю систему, которая взаимодействует с вашим приложением или системой. Актер - это своего рода тип.

– Прецедент: представляет действия, выполняемые одним или несколькими участниками в целях достижения определенной цели. Вариант использования - это своего рода тип.

– Цели: конечный результат большинства случаев использования.

Успешная диаграмма должна описывать действия и варианты, используемые для достижения цели.

Для построения взаимосвязей между акторами и прецедентами в языке UML имеется несколько стандартных видов отношений:

Отношение ассоциации (association relationship) - служит для обозначения специфической роли актера при его взаимодействии с отдельным вариантом использования.

– Отношение включения, включает в себя случай использования или вызывает включенный. Включение используется, чтобы показать, как случай использования разбивается на более мелкие шаги. Включенный вариант использования находится на конце наконечника стрелки.

– Отношение расширения (extend relationship) добавляет цели и шаги в расширенный вариант использования. Расширения работают только при определенных условиях. Расширенный вариант использования находится на конце наконечника стрелки.

– Отношение обобщения (generalization relationship) связывает специализированный и обобщенный элемент. Обобщенный элемент находится на конце стрелки. Специализированный вариант использования наследует цели и участников его обобщения и может добавлять более конкретные цели и шаги для их достижения.

При построении диаграммы вариантов использования хотелось показать, какие основные действия сможет совершать пользователь, такие как авторизоваться, внесение депозита, совершение ставки, вывод выигрыша. Все эти действия будут реализованы веб-сервисом. Общая диаграмма вариантов использования представлена на рисунке 7.

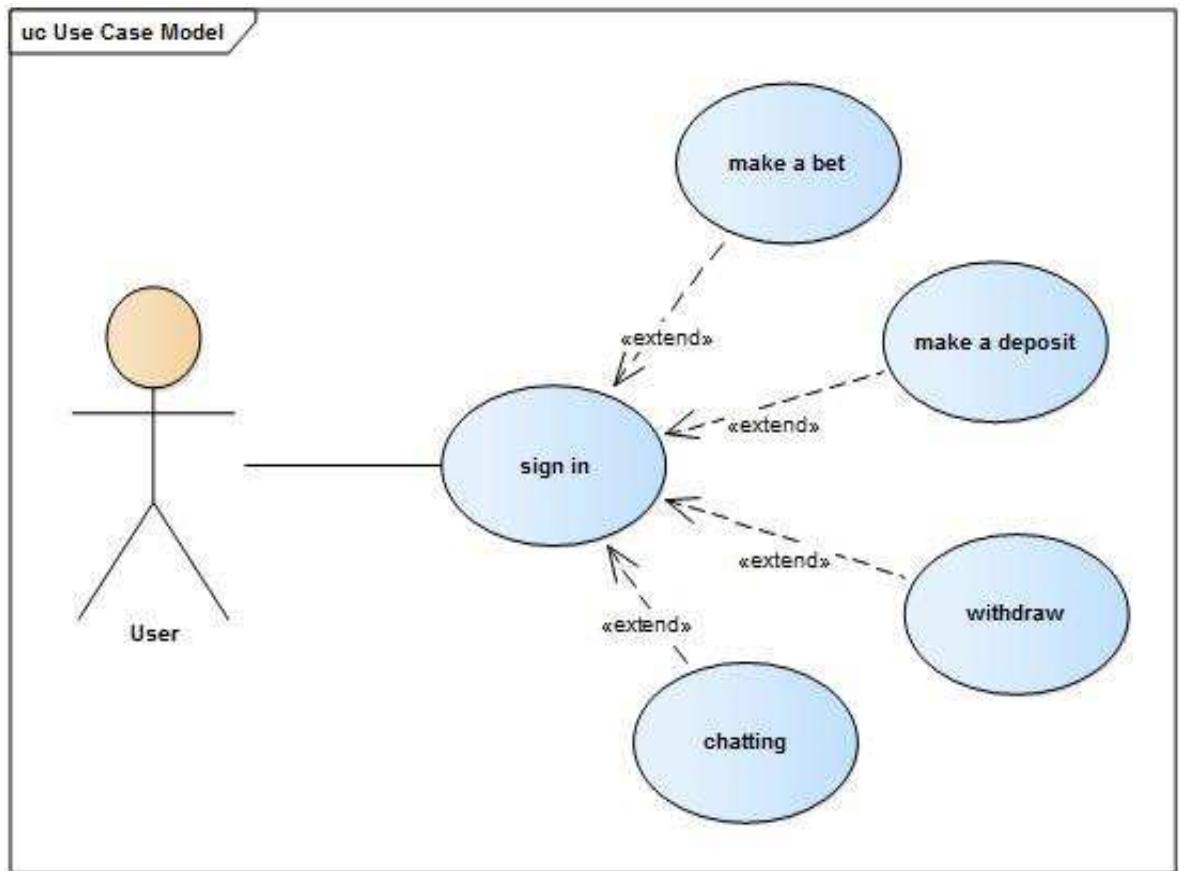


Рисунок 7 — Диаграмма вариантов использования

Конкретизация вариантов использования:

1 Sign in:

- Основное действующее лицо: Пользователь.
- Другие участники прецедента: отсутствуют.
- Связи с другими вариантами использования: отсутствуют.

– Краткое описание: данный вариант использования позволяет пользователю войти в систему для выполнения дальнейших действий, таких как: внесение депозита, участие в розыгрыше, использование чата, вывод предмета.

## 2 Make a deposit:

– Основное действующее лицо: Пользователь.  
– Другие участники прецедента: отсутствуют.  
– Связи с другими вариантами использования: отсутствуют.  
– Краткое описание: данный вариант использования позволяет пользователю обменять свои предметы на игровые поинты.

## 3 Make a bet:

– Основное действующее лицо: Пользователь.  
– Другие участники прецедента: отсутствуют.  
– Связи с другими вариантами использования: отсутствуют.  
– Краткое описание: данный вариант использования позволяет пользователю внести в игру определенное количество поинтов

## 4 Withdraw:

– Основное действующее лицо: Пользователь.  
– Другие участники прецедента: отсутствуют.  
– Связи с другими вариантами использования: отсутствуют.  
– Краткое описание: данный вариант использования позволяет пользователю вывести предметы к себе в инвентарь.

## 5 Chatting:

– Основное действующее лицо: Пользователь.  
– Другие участники прецедента: отсутствуют.  
– Связи с другими вариантами использования: отсутствуют.  
– Краткое описание: данный вариант использования позволяет пользователям коммуницировать друг с другом.

## **2 Проектирование веб-портала**

Чтобы упростить этап разработки было решено изначально спроектировать функциональную модель SADT, спроектировать базу данных и выбрать стек технологий.

### **2.1 SADT-диаграмма**

Методология SADT представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели системы какой-либо предметной области. Функциональная модель SADT отображает структуру процессов функционирования системы и ее отдельных подсистем, т. е. выполняемые ими действия и связи между этими действиями. Для этой цели строятся специальные модели, которые позволяют в наглядной форме представить последовательность определенных действий.

Исходными строительными блоками любой модели IDEF0 процесса являются деятельность (activity) и стрелки (arrows).

Для наглядности, и чтобы в дальнейшем не возникло проблем с разработкой было решено сначала построить SADT-диаграммы.

Контекстная диаграмма A0: специальный вид (контекстной) диаграммы IDEF0, состоящей из одного блока, описывающего функцию верхнего уровня, ее входы, выходы, управления, и механизм мы, вместе с формулировками цели модели и точки зрения, с которой строится модель.

На рисунке 8 показана диаграмма, описывающая функцию получения предмета.

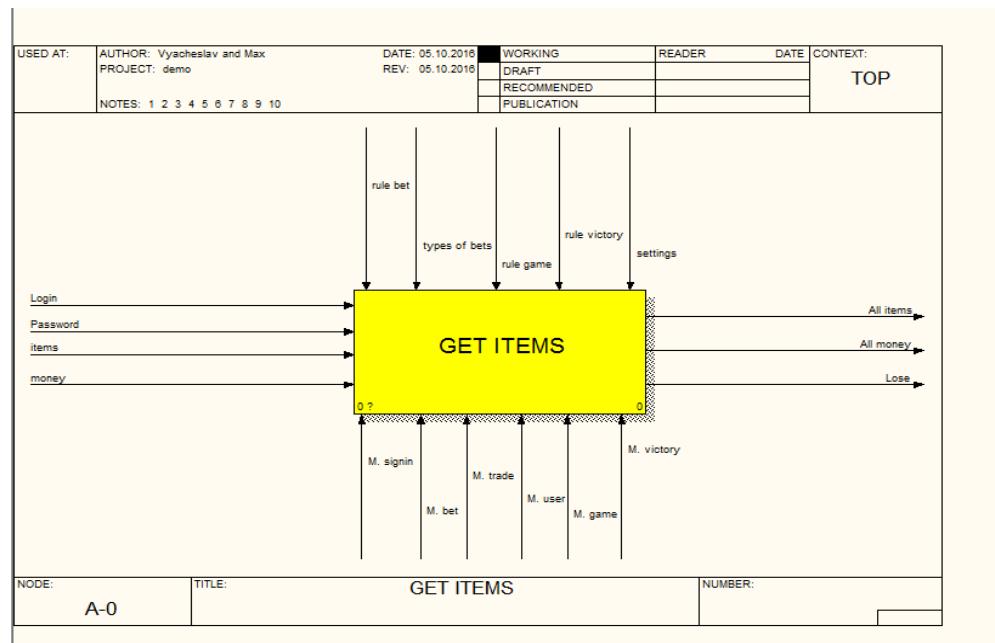


Рисунок 8 — Система получение предмета

На рисунке 9 представлена проведенная декомпозиция основной функции.

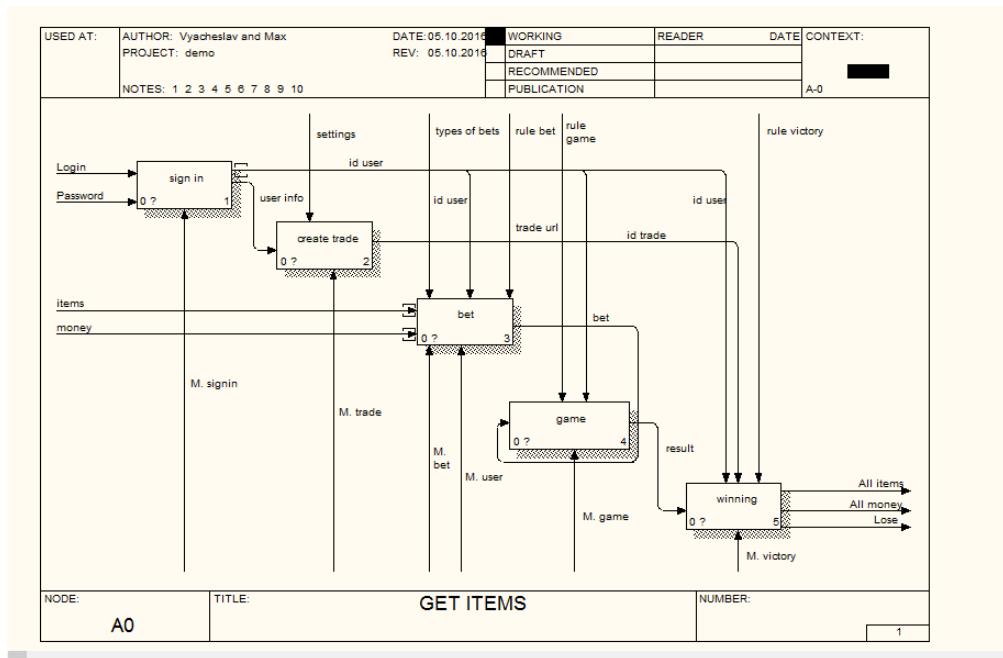


Рисунок 9 — Декомпозиция системы получение предмета

Система получения предмета состоит из следующих подсистем:

- sign in;
- create trade;
- bet;
- game;
- winning.

Диаграмма A1 – Авторизация, показана на рисунке 10.

A11(Авторизация через steam): пользователь через m.user переходит на форму авторизации через steam.

A12(Авторизация): Ввод логина и пароля.

A13(Подтверждение авторизации): Подтверждение, используя SteamGuard.

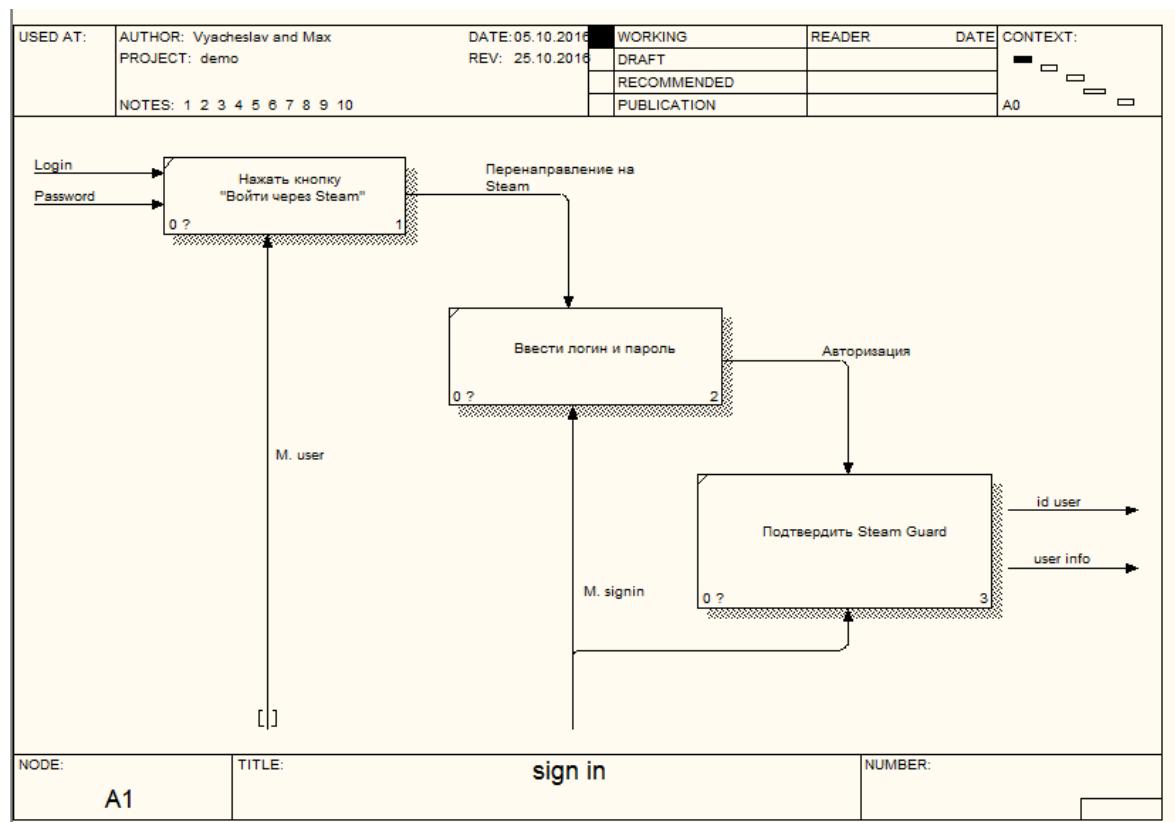


Рисунок 10 — Подсистема авторизации

### Диаграмма А3 – Create trade, показана на рисунке 11.

Пользователь должен перейти во вкладку настройка профиля, после, используя подсказки, добавить trade ссылку и нажать на кнопку сохранить изменения.

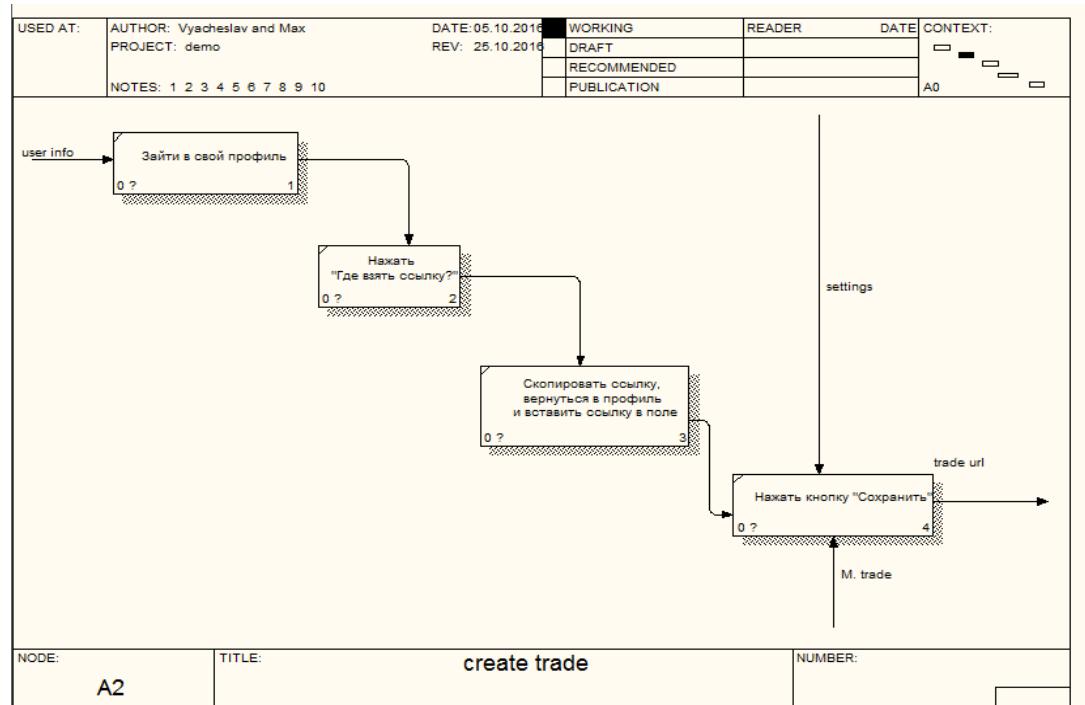


Рисунок 11 — Подсистема настройки аккаунта

### Диаграмма А4 – Bet, показана на рисунке 12.

A41(выбор способа ставки): пользователь через m.user выбирает интересующий его способ ставки.

A42(выбор способа оплаты): пользователь через m.user вносит ставку.

A43(Подтверждение ставки): пользователь через m.bet подтверждает ставку.

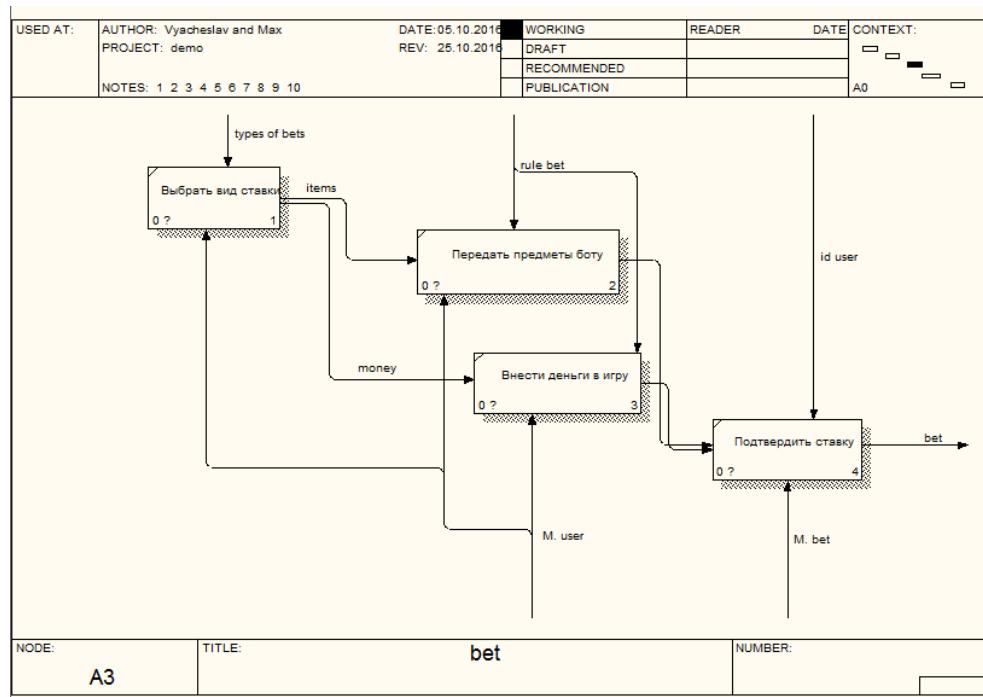


Рисунок 12 — Подсистема депозита

Диаграмма A5 –Game, показана на рисунке 13.

A51(принятие ставки)

A52(подсчет кол-ва игроков)

A53(запуск игры)

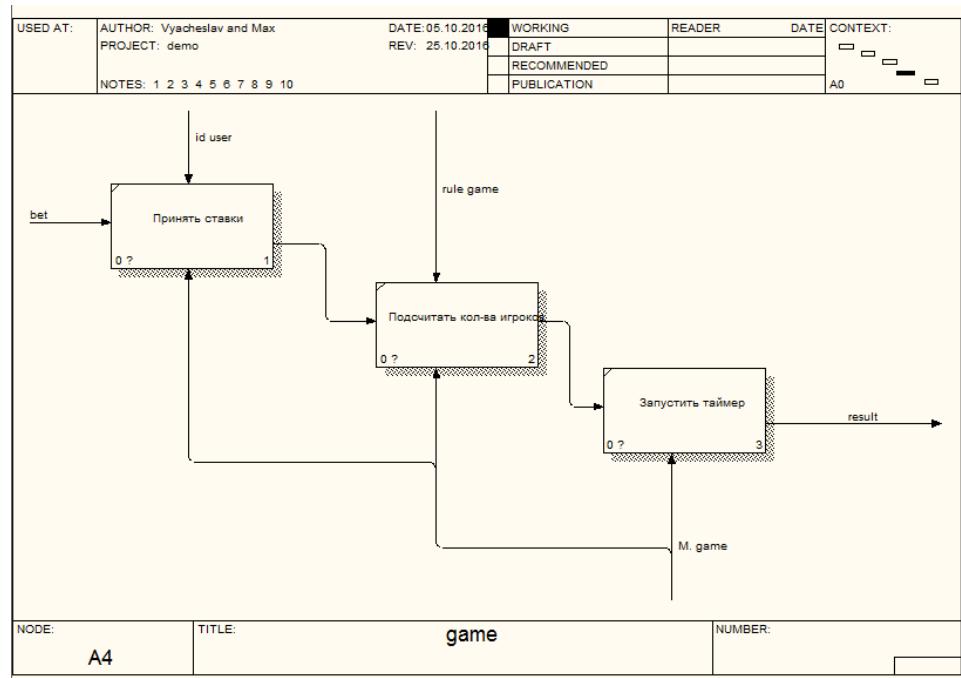


Рисунок 13— Подсистема игры

Диаграмма A6 –Game, показана на рисунке 14.

A61(результат игры)

A62(получение выигрыша)

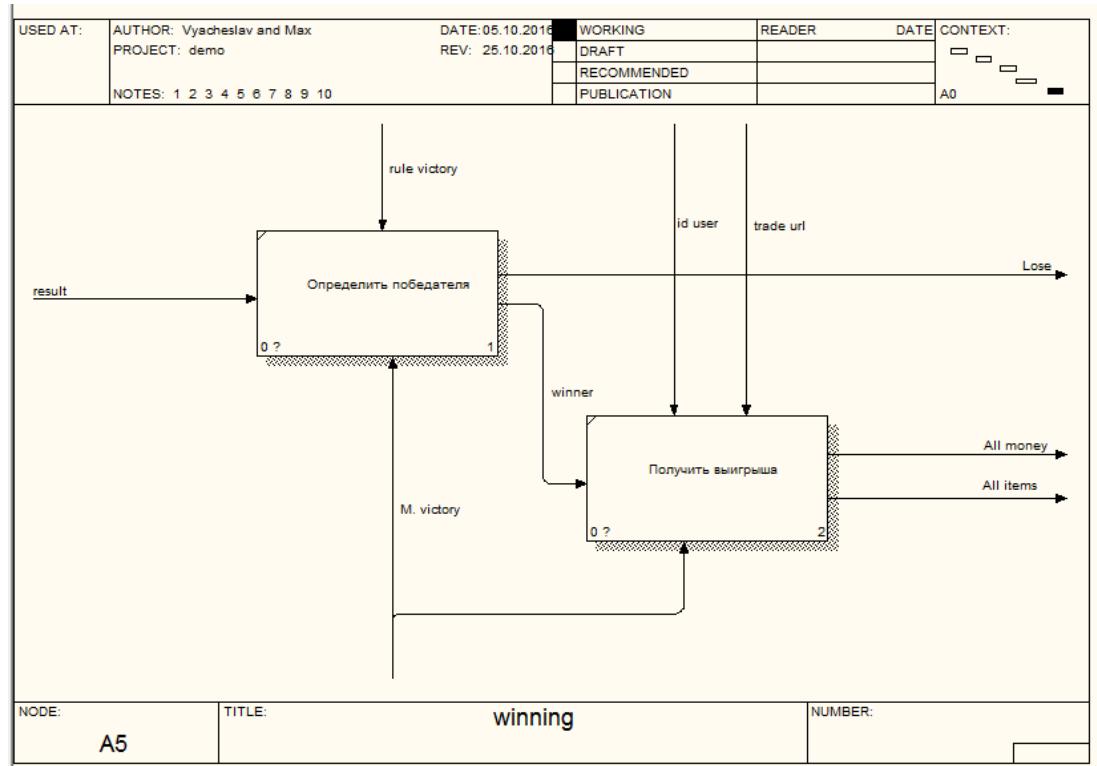


Рисунок 14 — Подсистема определение победителя

## **2.2 Проектирование базы данных**

Перед созданием сайта необходимо разработать структуру будущей базы данных. Используя программный продукт ErWinDataModeller, была спроектирована схема данных веб-сервиса.

Реляционная база данных – база данных, построенная на основе реляционной модели. В реляционной базе каждый объект задается записью (строкой) в таблице. Реляционная база создается и затем управляется с помощью реляционной системы управления базами данных. Фактически реляционная база данных это тело связанной информации, сохраняемой в двухмерных таблицах. Связь между таблицами может находить свое отражение в структуре данных, а может только подразумеваться, то есть присутствовать на неформализованном уровне. Каждая таблица БД представляется как совокупность строк и столбцов, где строки соответствуют экземпляру объекта, конкретному событию или явлению, а столбцы - атрибутам (признакам, характеристикам, параметрам) объекта, события, явления.

Реляционные базы данных предоставляют более простой доступ к оперативно составляемым отчетам (обычно через SQL) и обеспечивают повышенную надежность и целостность данных благодаря отсутствию избыточной информации.

На рисунке 15 представлена структура базы данных.

withdraws	
<b>id</b>	
offer id	
user id	
items	
value	
status	
date	
created at	
updated at	

settings	
<b>id</b>	
sitename	
descriptions	
metatags	
steam apikey	
socket port	
curs	
mrh ID	
order id	
mrh secret1	
mrh secret2	
ref user	
ref partner	
double timer	
double comission	
double max bet	
double min bet	
jackpot timer	
jackpot min bet	
jackpot max bet	
ticket curs	
jackpot maxitems	
jackpot max bet sum	
jackpot comission	
jackpot comission room2	
jackpot timer room2	
ticket curs room2	
jackpot maxitems room2	
jackpot max bet sum room2	
jackpot max bet room2	
jackpot min bet room2	
jackpot comission room3	
jackpot timer room3	
ticket curs room3	
jackpot maxitems room3	
jackpot max bet sum room3	
jackpot min bet room3	
matches comission	
matches min bet	
bonus sum	
dep minprice	
admin trade	
updated at	

users	
<b>id</b>	
steamid64	
IP	
username	
avatar	
money	
real money	
wonback money	
wagering money	
state	
trade	
is admin	
banchat	
ref id	
ref code	
activate code	
daily	
remember token	
created at	
updated at	

payments	
<b>id</b>	
secret	
merchant id	
order id	
sum	
user id	
status	
created at	
updated at	

Рисунок 15 — Схема данных

В таблице «withdraws» хранятся основные характеристики для вывода предмета.

В таблице «bots» хранится информация для бота.

В таблице «double» хранится подробная информация об игре.

В таблице «settings» хранятся настройки веб-сайта.

В таблице «user» хранится информация о пользователях.

В таблице «payments» храниться информация для внесения депозита.

## **2.3 Обзор выбранной архитектуры и инструментов для реализации веб-портала**

Для реализации данного проекта были выбраны следующие технологии: языки программирования PHP, JavaScript, framework Laravel, Bootstrap, веб-сервер Apache, база данных MySQL, а также программная платформа NodeJS.

Для удобства и быстроты современное сообщество используют готовый набор серверного программного обеспечения – LAMP, включающий в себя следующие компоненты:

- Linux – операционная система Linux;
- Apache – веб-сервер;
- MySQL – СУБД;
- PHP – язык программирования.

Изначально перечисленные программные продукты не разрабатывались специально для работы друг с другом, лишь в последствие данная комбинация стала весьма популярна в веб-разработке.

Apache — полнофункциональный, расширяемый веб-сервер с открытым кодом.

На сегодняшний день существуют огромное количество веб-серверов. Apache является самым популярным среди них, примерно на 70% серверов мира он используется.

Плюсы:

- Возможность простой аутентификации;
- Поддержка технологии SSI;
- Возможность создания на сервере пользовательских директорий;
- Возможность настройки виртуальных серверов;
- Работа с различными скриптами;
- Безопасность;
- Ведение журнала событий на сервере.

MySQL — это самая распространенная полноценная серверная СУБД. MySQL очень функциональная, свободно распространяемая СУБД, которая успешно работает с различными сайтами и веб приложениями.

Плюсы:

- Простота и легкость освоения.
- Большое сообщество пользователей и разработчиков.
- Богатый функционал.
- Скорость.
- Безопасность
- Масштабируемость.

PHP — известный язык программирования, с открытым исходным кодом, интенсивно применяющийся в веб-среде. Был разработан для создания скриптов работающих на сервере, так же обрабатывать данные html-форм динамически генерировать html-страницы и выполнения работ с различными базами данных.

Плюсы:

- Простота и легкость освоения.
- Работа с различными базами данных.
- Работа с различными веб-серверами.
- Большое сообщество пользователей и разработчиков.
- Имеется огромное количество библиотек и расширений языка.

Для удобства и быстроты разработки было решено использовать php-фреймворк.

На сегодняшний день создано очень большое количество фреймворков базирующиеся на php. Но в данной работе мы рассмотрим два laravel, codeigniter. Laravel — бесплатный веб-фреймворк с открытым кодом, использующиеся для разработки с использованием паттерна MVC. Пакеты — позволяют создавать и подключать модули в формате Composer к приложению на Laravel.

ORM (объектно-реляционное сопоставление) представляет собой расширенную реализацию PHP активного шаблона записи, предоставляя в то же время внутренние методы для ограничений на отношения между объектами бд.

После активного шаблона записи Eloquent ORM представляет таблицы базы данных как классы, а их экземпляры объектов привязаны к одиночным строкам таблицы.

Создатель запросов, доступный с Laravel 3, обеспечивает более прямой доступ к базе данных, альтернативный Eloquent ORM. Вместо того, чтобы требовать непосредственного написания SQL-запросов, построитель запросов Laravel предоставляет набор классов и методов, способных создавать запросы.

Логика приложения является неотъемлемой частью разработанных приложений, реализованных либо с использованием контроллеров, либо как часть деклараций маршрутов. Синтаксис используется для определения логики приложения аналогична той, которая используется Синатра рамок.

Обратная маршрутизация определяет взаимосвязь между ссылками и маршрутами, позволяя автоматически изменять маршруты в соответствующие ссылки. Когда ссылки создаются с использованием имен существующих маршрутов, подходящие унифицированные идентификаторы ресурсов (URI) автоматически создаются Laravel.

Остальные контроллеры предоставляют дополнительный способ разделения логики обслуживания HTTP GET и POST-запросов.

Автоматическая загрузка класса обеспечивает автоматическую загрузку классов PHP без необходимости ручного обслуживания путей включения. Загрузка по требованию предотвращает включение ненужных компонентов, поэтому загружаются только фактически используемые компоненты.

Представленные композиторы представляют собой настраиваемые логические единицы кода, которые могут выполняться при загрузке представления.

Автоматическая разбивка на страницы упрощает задачу реализации разбивки на страницы, заменяя обычные ручные реализации с автоматизированными методами, интегрированными в Laravel.

CodeIgniter — популярный MVC фреймворк с открытым исходным кодом, написанный на языке программирования PHP, для разработки полноценных веб-систем и приложений.

Поддержка баз данных MySQL, PostgreSQL, MSSQL, SQLite, Oracle.

Поддержка псевдо-ActiveRecord, который по большей части повторяет синтаксис языка SQL

Легко расширяемая система за счет возможности использования сторонних и самописных библиотек, а также дополнения или переопределения существующих.

Поддержка как сегментированных ЧПУ, так и обычных URL-ов с передачей параметров.

Фреймворк содержит в себе множество необходимых библиотек, которые создают функциональность для работы с файлами, отправки электронных писем, валидации форм, поддержки сессий, работы с изображениями и так далее.

Основываясь на вышеписанном и мнении комьюнити был сделан вывод, что Laravel обладает большей гибкостью, расширяемостью, безопасностью и лучшей производительностью. Исходя из этих плюсов было решено использовать в рамках нашего проекта framework laravel.

Выбор framework laravel в свою очередь определил архитектуру проектирования model-view-controller.

Основная цель данного паттерна заключается в разделение приложения на три уровня: модель данных (model), представление (view) и контроллер (controller). Таким образом, изменения, вносимые в один из компонентов, оказывают минимальное возможное воздействие на другие компоненты или не оказывают вовсе. Рассмотрим этот паттерн подробнее.

Model (модель) – это часть, в которой содержится бизнес-логика приложения. Она никак не зависит от остальных частей. Модельный слой ничего не должен знать об элементах дизайна (View), и каким образом он будет отображаться.

**View** (вид, представление) – данная часть отвечает за визуализацию данных, полученных из контроллера и модели. Так, например, одни и те же данные могут представляться разными способами. Она содержит HTML-разметку и небольшие вставки PHP-кода.

**Controller** (контроллер) — часть, связывающая модели, виды и другие компоненты в работающие приложение. То есть обрабатывает поступающие запросы, выполняет операции с моделью и выбирает представления для визуализации пользователю.

На рисунке 16 показано взаимодействие между уровнями приложения.



Рисунок 16 — Model-view-controller

Так как в проекте должна была присутствовать интерактивность встал вопрос использовать чистый JavaScript или библиотеку jQuery, для понимания jQuery является библиотекой JavaScript. Основные отличительные свойства представлены в таблице 1

Таблица 1 – JS vs jQuery

JavaScrip	jQuery
Слабо типизированный язык динамического программирования	Быстрая и сжатая библиотека JavaScript
Язык сценариев взаимодействия интерфейса и управления содержимым документа	Рамка, которая упрощает и ускоряет обработку событий, анимацию и Ajax
Интерпретируемый язык	Использует ресурсы, предоставленные JavaScript, чтобы упростить работу
Нужно писать собственные сценарии, которые могут занять время	не нужно писать много скриптов, которые уже существуют в jQuery
Разработчики должны обрабатывать совместимость с несколькими браузерами, написав собственный код JavaScript	Является многосерверной библиотекой JavaScript, которая уменьшает работу разработчиков во время развертывания
Разработчики склонны делать много распространенных ошибок, связанных с браузером.	Разработчикам не нужно беспокоиться о проблемах совместимости браузеров.
Не нужно ничего включать в браузер, поскольку все современные браузеры поддерживают JS	Необходимо включить URL-адрес библиотеки jQuery в заголовок страницы
Больше строк кода	Меньше строк кода
Быстрее в доступе к DOM	Подходит для сложных операций, в которых разработчики склонны к ошибкам и написанию плохих строк кода.

Таким образом выбор пал на jQuery, т.к хорошо подходит для большинства приложений, которые требуют быстроты действий. jQuery заботится об общих ошибках браузера, а также заботится о проблеме совместимости браузера с программным продуктом.

Чтобы веб-сервис обладал адаптивностью было решено при разработке front-end использовать framework bootstrap

Framework bootstrap это простой и легко настраиваемый HTML, CSS и Javascript фреймворк для более быстрой и удобной разработки.

Включает в себя HTML и CSS-шаблоны оформления для типографики, веб-форм, кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса, включая JavaScript-расширения.

Основные инструменты:

Сетки - это заранее заданные размеры колонок, которые можно сразу применить. Сетка включает в себя 12 колонок для различных девайсов, что позволяет веб-странице быть масштабируемой, в результате чего создаётся адаптивный дизайн веб-приложения.

Шаблоны - это фиксированный или резиновый шаблон документа.

Формы - это классы для оформления форм и некоторых событий, происходящих с ними.

Медиа - представляет некоторое управление изображениями и видео.

Таблицы - это средства оформления таблиц, вплоть до добавления функциональности сортировки.

Типографика - это описание шрифтов, определение некоторых классов для шрифтов.

Навигация является классами оформления для табов, вкладок, страничности (пагинации), меню и панели инструментов.

Алерты – это оформление подсказок, диалоговых и всплывающих окон.

Все эти инструменты помогают настроить Bootstrap индивидуально под каждый проект, веб-сайт или веб-приложение.

Так же в рамках проекта нужно было реализовать программу, имитирующую деятельность человека. Так как, одной из основных функций было взаимодействие с API steam. При рассмотрение данной задачи было выявлено, что данную программу можно реализовать двумя способами, использовав python или NodeJS.

Но так как в нашем проекте за основу был взят framework Laravel, написанный на php, то использовать python не рационально. Поэтому выбор пал на NodeJS.

Node.js — программная платформа, на основе движка JavaScript Chrome V8, превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API, подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера, но есть возможность разрабатывать на Node.js и десктопные оконные приложения.

### 3 Разработка веб-сервиса

Для серверной части на понадобится установить веб-сервер, который имеет все необходимое для работы с фреймворком. Устанавливаем менеджер зависимостей для PHP Composer, который достаточно просто установить для Windows. Далее с помощью командной строки устанавливаем Larvael командой composer global require "laravel/installer" и создаем проект laravel new reg. На рисунке 17 представлена структура проекта.

Имя	Размер	Изменено	Права	Владел
..		18.05.2018 13:04:22	rwxr-xr-x	root
app		27.06.2017 16:25:26	rwxr-xr-x	root
bootstrap		18.06.2017 18:06:25	rwxrwxrwx	root
config		18.06.2017 18:06:28	rwxr-xr-x	root
database		18.06.2017 18:06:25	rwxr-xr-x	root
public		27.01.2018 3:28:52	rwxr-xr-x	root
resources		18.06.2017 18:06:25	rwxr-xr-x	root
storage		18.06.2017 18:06:28	rwxrwxrwx	root
tests		18.06.2017 18:06:27	rwxr-xr-x	root
vendor		18.06.2017 18:06:27	rwxr-xr-x	root
.env	1 KB	27.01.2018 3:03:53	rw-r--r--	root
.gitattributes	1 KB	18.06.2017 18:06:25	rw-r--r--	root
.gitignore	1 KB	18.06.2017 18:06:25	rw-r--r--	root
.htaccess1	1 KB	18.06.2017 18:06:25	rw-r--r--	root
artisan	2 KB	18.06.2017 18:06:25	rw-r--r--	root
composer.json	2 KB	18.06.2017 18:06:25	rw-r--r--	root
composer.lock	134 KB	18.06.2017 18:06:25	rw-r--r--	root
gulpfile.js	1 KB	18.06.2017 18:06:25	rw-r--r--	root
package.json	1 KB	18.06.2017 18:06:25	rw-r--r--	root
phpspec.yml	1 KB	18.06.2017 18:06:25	rw-r--r--	root
phpunit.xml	1 KB	18.06.2017 18:06:25	rw-r--r--	root
readme.md	2 KB	18.06.2017 18:06:25	rw-r--r--	root
server.php	1 KB	18.06.2017 18:06:25	rw-r--r--	root

Рисунок 17 — Структура проекта

Точка входа, иначе говоря, index.php, располагается в каталоге public. По задумке разработчиков фреймворка в каталоге public хранятся файлы, к которым разрешен публичный доступ. Для сохранения структуры проекта прописываем в файле .htaccess правила RewriteEngine On, RewriteRule ^\$ public/ [L], RewriteRule (.\*) public/\$1 [L], то есть все запросы перенаправляются в каталог public.

В каталоге app располагаются основная логика приложения, например, модели, контроллеры и так далее. Непосредственно в корне каталога app располагаются модели user.php (создана по умолчанию). В каталоге app/http/controllers хранятся контроллеры. На рисунке 18 представлена основная логика приложения.

/var/www/coin/app					
Имя	Размер	Изменено	Права	Владел	
..		21.05.2018 16:38:12	rwxr-xr-x	root	
Console		18.06.2017 18:06:28	rwxr-xr-x	root	
Events		18.06.2017 18:06:28	rwxr-xr-x	root	
Exceptions		18.06.2017 18:06:28	rwxr-xr-x	root	
Http		18.06.2017 18:06:28	rwxr-xr-x	root	
Jobs		18.06.2017 18:06:28	rwxr-xr-x	root	
Listeners		18.06.2017 18:06:28	rwxr-xr-x	root	
Providers		18.06.2017 18:06:28	rwxr-xr-x	root	
Deposits.php	1 KB	18.06.2017 18:06:28	rw-r--r--	root	
Double.php	1 KB	18.06.2017 18:06:28	rw-r--r--	root	
Settings.php	1 KB	18.06.2017 18:06:28	rw-r--r--	root	
User.php	1 KB	18.06.2017 18:06:28	rw-r--r--	root	

Рисунок 18 — Основная логика приложения

В каталоге resources/view находятся виды, которые переведены в php с использование шаблонизатора Blade. Все шаблоны компилируются в PHP код и кешируются до следующих изменений, что означает — Blade не добавляет накладных расходов в приложение. Два основных преимущества Blade: наследование и секции. Основные view представлены на рисунке 19.

Имя	Размер	Изменено	Права	Владел
..		18.06.2017 18:06:25	rwxr-xr-x	root
admin		27.06.2017 16:24:34	rwxr-xr-x	root
errors		18.06.2017 18:06:25	rwxr-xr-x	root
includes		24.06.2017 17:50:31	rwxr-xr-x	root
pages		29.06.2017 15:14:06	rwxr-xr-x	root
vendor		18.06.2017 18:06:25	rwxr-xr-x	root
admin.blade.php	7 KB	18.06.2017 18:06:25	rw-r--r--	root
layout.blade.php	11 KB	29.06.2017 15:12:54	rw-r--r--	root

Рисунок 19 — Views

На рисунке 20 показаны контроллеры, отвечающие за логику веб-сервиса. Основные из них, DepositController, ChatController, DoubleController, AuthController, ShopController.

Имя	Размер	Изменено	Права	Владел
..		18.06.2017 18:06:28	rwxr-xr-x	root
AdminController.php	21 KB	27.06.2017 18:29:18	rw-r--r--	root
ApiController.php	1 KB	08.06.2017 18:22:07	rw-r--r--	root
AuthController.php	4 KB	02.03.2017 7:23:40	rw-r--r--	root
ChatController.php	8 KB	25.06.2017 1:57:49	rw-r--r--	root
Controller.php	1 KB	02.02.2017 20:15:36	rw-r--r--	root
DepositController.php	7 KB	27.06.2017 16:08:09	rw-r--r--	root
DoubleController.php	24 KB	27.06.2017 17:23:13	rw-r--r--	root
LanguageController.p	1 KB	24.02.2017 6:43:02	rw-r--r--	root
PagesController.php	15 KB	29.06.2017 15:13:55	rw-r--r--	root
ShopController.php	11 KB	27.06.2017 17:47:46	rw-r--r--	root
words.json	3 KB	01.03.2017 6:33:20	rw-r--r--	root

Рисунок 20 — Контроллеры

Логика контроллера, отвечающего за внесение депозита показана на рисунке 21.

```
class DepositController extends Controller

    public function __construct()
    {
        $this->redis = Redis::connection();
        $this->config = Settings::first();
        if(Auth::check()) {
            $this->user = Auth::user();
            view()->share('u', $this->user);
        }
        $this->prices = json_decode(File::get('prices.txt'), true);
        view()->share('config', $this->config);
    }

    public function index()
    {
        view()->share('title', 'Deposit');
        return view('pages.refill');
    }

    public function passItems()
    {
        if(Auth::guest()) return;

        $res = json_decode(file_get_contents('http://steamcommunity.com/inventory/'. $this->user->steamid64 . '/730/2'));
        if(!isset($res->assets)) return;

        $items = [];
        foreach($res->assets as $item) {
            $items[] = [
                'classid'      => $item->classid,
                'instanceid'   => $item->instanceid,
                'assetid'      => $item->assetid
            ];
        }

        foreach($items as $i => $item) {
            $found = false;
            foreach($res->descriptions as $data) {
                if(!$found) && ($data->classid == $item['classid']) && ($data->instanceid == $item['instanceid']) {
                    $found = true;
                    $items[$i]['market_hash_name'] = $data->market_hash_name;
                }
            }
        }

        $list = [];
        foreach($items as $i => $item) {
            if(isset($this->prices[$item['market_hash_name']])) {
                $items[$i]['price'] = floor($this->prices[$item['market_hash_name']] * $this->config->curs);
                $list[] = $items[$i];
            }
        }
    }
}
```

Рисунок 21 — Логика контроллера

Логика контроллера, отвечающего за чат показана на рисунке 22.

```
class ChatController extends Controller
{
    const CHAT_CHANNEL = 'chat.message';
    const NEW_MSG_CHANNEL = 'new.msg';
    const DELETE_MSG_CHANNEL = 'del.msg';

    public function __construct()
    {
        parent::__construct();
        $this->redis = Redis::connection();
    }

    public function banchat(Request $request) {
        $steamid64 = $request->get('steamid64');
        $user = User::where('steamid64', $steamid64)->first();

        if($user->banchat == 0){
            User::where('steamid64', $user->steamid64)->update(['banchat' => '1']);
            return response()->json(['status' => 'success', 'reason' => 'Пользователь заблокирован.']);
        }else{
            return response()->json(['status' => 'error', 'reason' => 'Пользователь уже заблокирован.']);
        }
    }

    public function unbanchat(Request $request) {
        $steamid64 = $request->get('steamid64');
        $user = User::where('steamid64', $steamid64)->first();

        if($user->banchat == 1){
            User::where('steamid64', $user->steamid64)->update(['banchat' => '0']);
            return response()->json(['status' => 'success', 'reason' => 'Пользователь разблокирован.']);
        }else{
            return response()->json(['status' => 'error', 'reason' => 'Пользователь не заблокирован.']);
        }
    }

    public static function chat()
    {
        $redis = Redis::connection();

        $value = $redis->lrange(self::CHAT_CHANNEL, 0, -1);
        $i = 0;
        $returnValue = NULL;
        $value = array_reverse($value);

        foreach ($value as $key => $newchat[$i]) {
            if ($i > 20) {
                break;
            }
            $value2[$i] = json_decode($newchat[$i], true);

            $user = DB::table('users')->where('steamid64', $value2[$i]['steamid64'])->first();

            $value2[$i]['username'] = htmlspecialchars($value2[$i]['username']);

            $returnValue[$i] = [
                'userid' => $value2[$i]['userid'],
                'steamid64' => $value2[$i]['steamid64'],
                'avatar' => $value2[$i]['avatar'],
                'colors' => $value2[$i]['colors'],
                'time' => $value2[$i]['time'],
                'time2' => $value2[$i]['time2'],
                'support' => $value2[$i]['support'],
                'ban' => $value2[$i]['ban'],
                'mute' => $user->banchat,
                'messages' => $value2[$i]['messages'],
                'username' => $value2[$i]['username'],
                'admin' => $value2[$i]['admin'],
            ];
        }
        $i++;
    }
}
```

Рисунок 22 — Логика контроллера

Функция, отвечающая за внесение поинтов в игру показана на рисунке 23.

```
public function addBet(Request $r)
{
    if(Auth::guest()) return;

    if(Session::get('applocale') == 'ru') {
        if($r->get('value') < $this->config->double_min_bet) return [
            'msg'      => 'Минимальная сумма ставки - '.$this->config->double_min_bet,
            'type'     => 'error'
        ];

        if($r->get('value') > $this->config->double_max_bet) return [
            'msg'      => 'Максимальная сумма ставки - '.$this->config->double_max_bet,
            'type'     => 'error'
        ];
    }

    if ($r->get('real') == 'false')
    {
        if($this->user->money < $r->get('value')) return [
            'msg'      => 'Недостаточно Free Coin на балансе!',
            'type'     => 'error'
        ];
    }
    else
    {
        if($this->user->real_money < $r->get('value')) return [
            'msg'      => 'Недостаточно Real Coin на балансе!',
            'type'     => 'error'
        ];
    }

    if($this->game->status > 1) return [
        'msg'      => 'Ставки в эту игру уже не принимаются!',
        'type'     => 'error'
    ];
} else {
    if($r->get('value') < $this->config->double_min_bet) return [
        'msg'      => 'The minimum bet amount - '.$this->config->double_min_bet,
        'type'     => 'error'
    ];

    if($r->get('value') > $this->config->double_max_bet) return [
        'msg'      => 'The maximum bet amount - '.$this->config->double_max_bet,
        'type'     => 'error'
    ];

    if($this->user->money < $r->get('value')) return [
        'msg'      => 'Not enough PT in the balance!',
        'type'     => 'error'
    ];

    if($this->game->status > 1) return [
        'msg'      => 'Bets in this game are no longer accepted!',
        'type'     => 'error'
    ];
}
```

Рисунок 23 — Функция «addBet»

Для автоматизации процессов на сайте был реализован бот на nodeJS с использованием специальной библиотекой steam. Изначально настройки веб-сервиса мы получаем напрямую из бд, соединив бота с субд, данная функция показана на рисунке 24.

```
C:\Users\alans\Dropbox\paper_examples-master\README |  
var config = require('./config.js'),  
app = require('express')(),  
server = require('http').Server(app),  
io = require('socket.io')(server);  
  
/* - Dev - */  
var double_timer = 0,  
rotate = 0;  
  
var online = 0;  
  
var timer, ngtimer;  
  
/* - Log Function - */  
function log(log) { console.log('[APP] ' + log); }  
  
/* - Etc. init - */  
  
var mysql = require('mysql');  
  
var client = mysql.createConnection({  
host : config.mysql.host,  
port : config.mysql.port,  
user : config.mysql.user,  
password : config.mysql.password,  
database : config.mysql.database  
});  
  
client.query('select * from settings', function(err, res) {  
if(err) {  
log('Ошибка при выполнении mysql запроса.');//  
return;  
}  
if(typeof res[0] == 'undefined') {  
log('Не удалось найти настройки сайта.');//  
} else {  
res = res[0];  
server.listen(res.socket_port);  
log('Socket сервер запущен на порте ' + res.socket_port);  
}  
};
```

Рисунок 24 — Соединение бота с БД

После бот в автоматическом режиме контролирует процесс игры, основные из них: проверяет ставки, начинает игру, определяет победителя, создает новую игру, выводит предметы

Функция внесения депозита показана на рисунке 25.

```
function deposit(data) {
if(!cookiesSet) return setTimeout(function() {
deposit(data);
}, 5000);
var sendItems = [];
for(var i = 0; i < data.items.length; i++) {
if(typeof data.items[i] != 'undefined') sendItems.push({
appid : 730,
contextid : 2,
assetid : data.items[i]
});
}
var offer = Manager.createOffer(data.url);
offer.addTheirItems(sendItems);
sendDepOffer(offer, data, 5);
}
```

Рисунок 25 — Функция внесения депозита

Функция вывода предметов показана на рисунке 26.

```
function withdraw(data) {
if(!cookiesSet) return setTimeout(function() {
withdraw(data);
}, 5000);
var sendItems = [];
for(var i = 0; i < data.items.length; i++) {
if(typeof data.items[i] != 'undefined') sendItems.push({
appid : 730,
contextid : 2,
assetid : data.items[i]
});
}
var offer = Manager.createOffer(data.url);
offer.addMyItems(sendItems);
offer.setMessage('Выход с сайта ' + config.siteName + ' на сумму ' + data.value + ' PTS');
sendOffer(offer, data, 5);
}
```

Рисунок 26 — Функция вывода предметов

Функция авторизации показана на рисунке 27.

```
function login(data) {
  Steam.login({
    accountName : data.username,
    password : data.password,
    twoFactorCode : SteamTotp.generateAuthCode(data.shared_secret)
  }, function(err, sessionID, cookies) {
    if(err) {
      log('Ошибка при авторизации в STEAM. Повторяем через 10 сек. ('+err.message+')');
      setTimeout(function() {
        login(data);
      }, 10000);
      return;
    }
    log('Авторизировались в STEAM!');
    Manager.setCookies(cookies, function(err) {
      if(err) {
        log('Ошибка при подключении куки к SteamTradeofferManager. Переходим в STEAM через 10 сек.');
        setTimeout(function() {
          login(data);
        }, 10000);
        return;
      }
      log('Подключили куки к SteamTradeofferManager!');
      cookiesSet = true;
    });
    Steam.startConfirmationChecker(10000, data.identity_secret);
  });

  Steam.on('confKeyNeeded', function(tag, callback) {
    var time = Math.floor(Date.now() / 1000);
    callback(null, time, SteamTotp.getConfirmationKey(data.identity_secret, time, tag));
  });

  Steam.on('newConfirmation', function(confirmation) {
    var time = Math.floor(Date.now() / 1000);
    var key = SteamTotp.getConfirmationKey(data.identity_secret, time, 'allow');
    confirmation.respond(time, key, true, function(err) {
      if(err) {
        log(err);
        return;
      }
      log('Успешно подтвердили трейд');
    });
  });
}

Manager.on('newOffer', function(offer) {
  if(offer.itemsToReceive.length !== 0 && config.admins.indexOf(offer.partner.getSteamID64()) != -1) {
    offer.accept(function (err, status) {
      if(!err) {
        log("Трейд от админа #" + offer.id + " Принят");
      } else {
        log("Трейд от админа #" + offer.id + " Ошибка");
      }
    });
  }
  if(offer.itemsToGive.length !== 0) {
    offer.decline();
  }
});
```

Рисунок 28 — Функция авторизации

Для хранения информации была создана база данных, структура базы данных показана на рисунке 29.

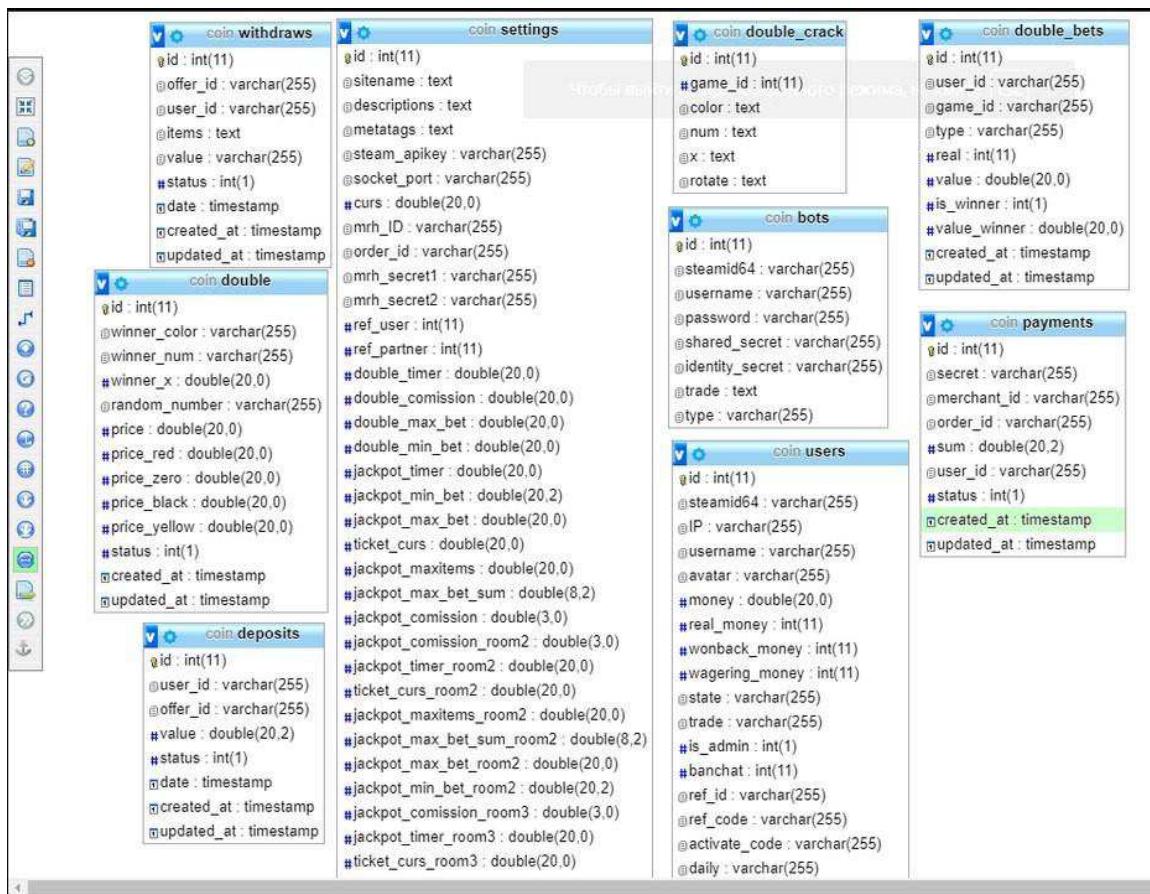


Рисунок 29 — Схема данных БД

Главной особенностью в созданной базе данных является отсутствие связей между таблицами. Данное решение было принято исходя из двух пунктов. Первый пункт заключается в том, что изначально базы данных проектировались без связей между таблицами, позже было решено реализовать возможность построения связей, одной из причин является каскадное удаление данных, т.к объемы памяти были малы и не позволяли хранить большой объем данных, но на сегодняшний день такой проблемы нет. Второй пункт заключается в том, что существующие связи между таблицами существенно затормаживают систему, поэтому было решено отказаться от их построения, что привело к более быстрой работе системы.

Интерфейс сайта создавался на основе готового макета с использованием framework bootstrap 3, использование готового макета позволило существенно сэкономить время разработки конечного продукта.

## 4 Апробация веб-портала (на примере действий пользователей)

В данной главе будут рассмотрены результаты основных действий пользователей.

При авторизации пользователь попадает на сайт steam, где вводит логин и пароль, при этом подтверждает, что его данные аккаунта будут публичны, после этого он перенаправляется на наш веб-портал. Форма авторизации продемонстрирована на рисунке 30.

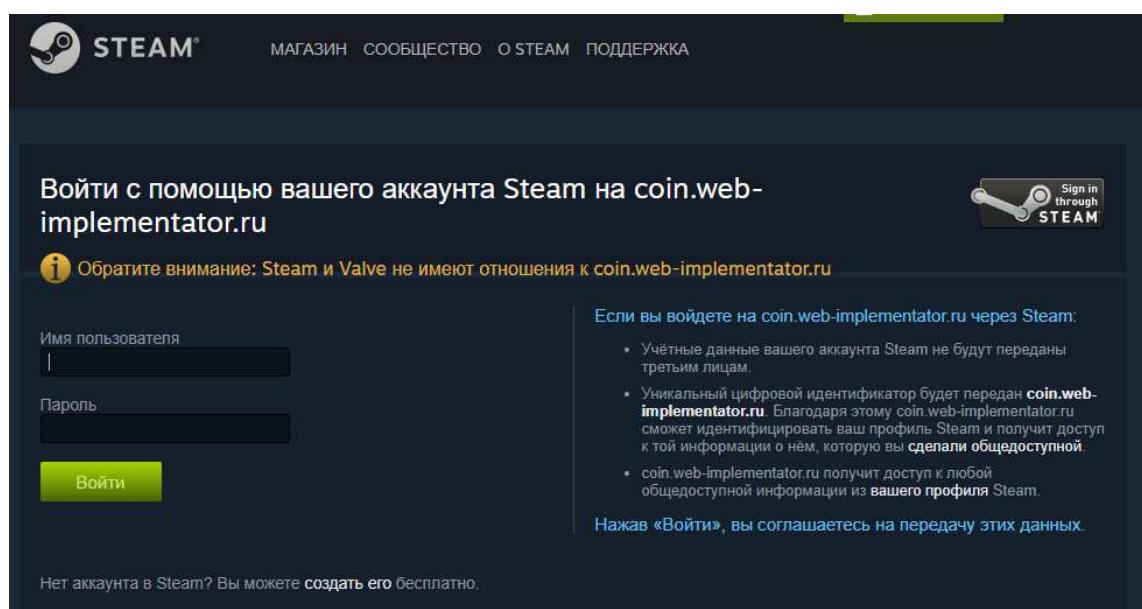


Рисунок 30 — Форма авторизация

Для внесений предметов пользователь должен перейти на вкладку «deposit». После перехода сперва идёт загрузка инвентаря аккаунта. После того как загрузка окончена, пользователь выбирает предметы для обмена на поинты и нажимает депозит. При этом бот отправляет данный депозит на подтверждение. После того как пользователь подтвердил операцию у него на счету появляется free coin и он может принимать участие в розыгрыше. Данная форма продемонстрирована на рисунке 31.

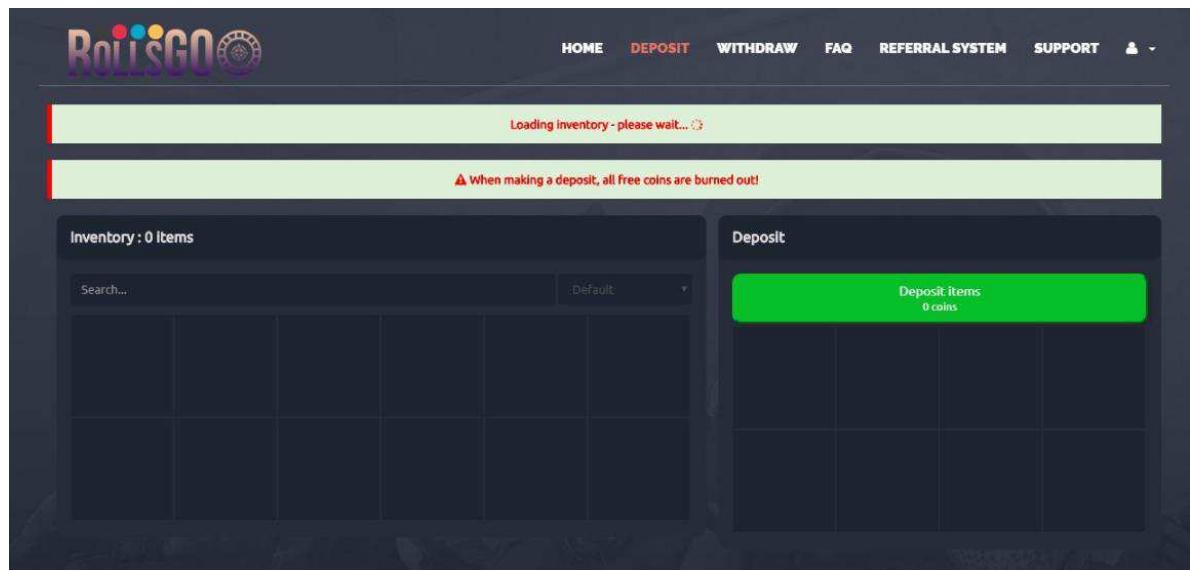


Рисунок 31 — Форма внесения депозита

На данном веб-портале используется следующий алгоритм, при котором пользователю доступны 4 цвета, он выбирает один из них и проводится розыгрыш, если участник угадывает, то его выигрыш зависит от выбранного им цвета. На рисунке 32 продемонстрирована данная процедура.

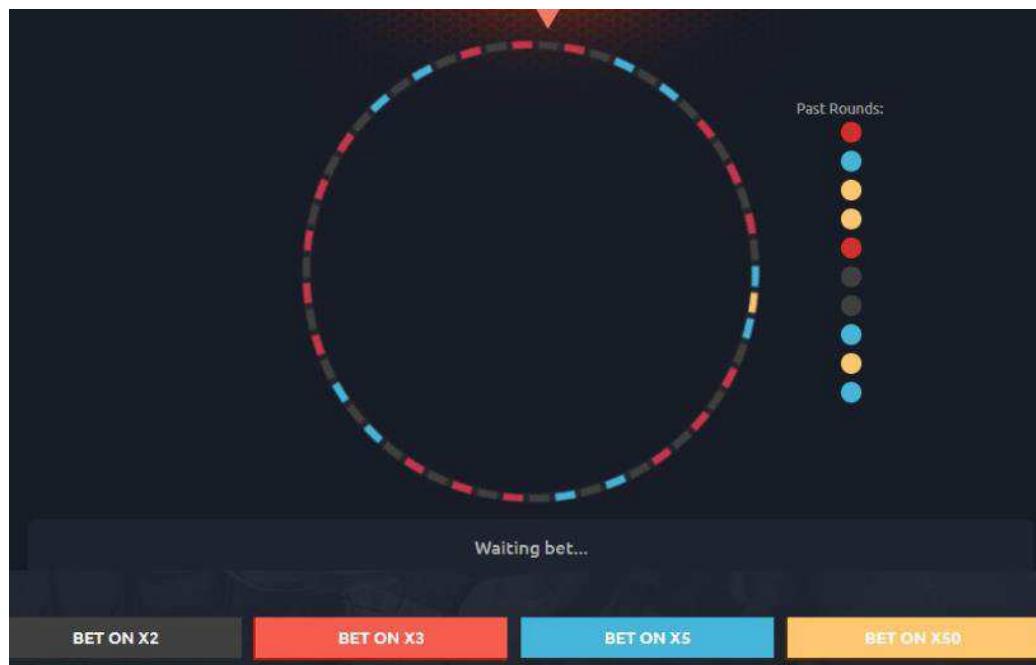


Рисунок 32 — Процедура игры

После выигрыша пользователь может обменять free coin на real coin, для того чтобы можно было совершать покупку игровых товаров. Данная процедура показана на рисунке 33.



Рисунок 33 — Функция обмена free coin на real coin

Передача игровых предметов на аккаунт пользователя проходит на вкладке «withdraw». После перехода по ней, юзер выбирает интересующие его предметы и подтверждает покупку, после этого бот автоматически передает скин в инвентарь пользователя. Функция покупки предмета продемонстрирована на рисунке 34.

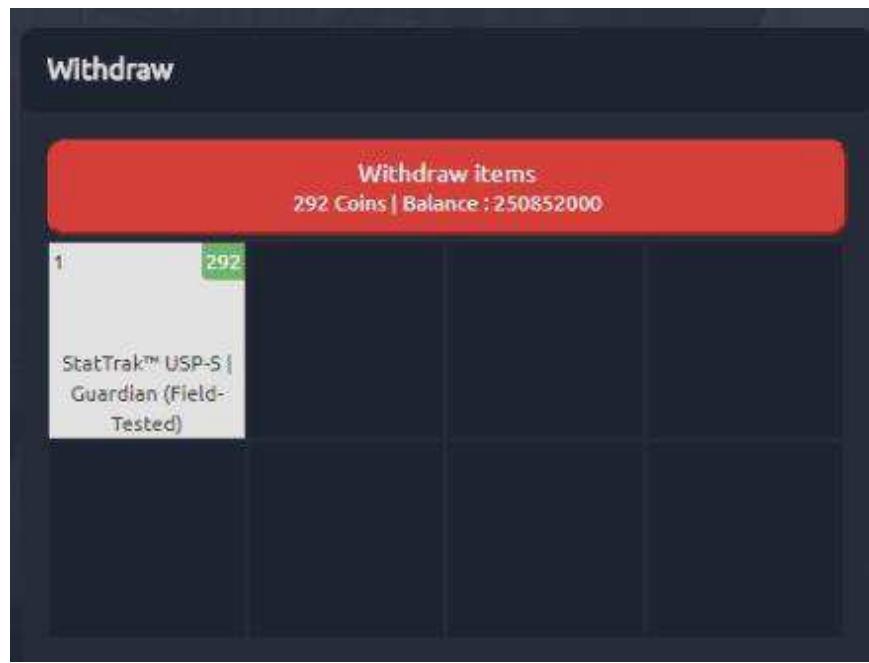


Рисунок 34 — Функция вывода предметов

Так же была реализована функция «чат», чтобы игроки могли коммуницировать друг с другом, данная функция показана на рисунке 35.

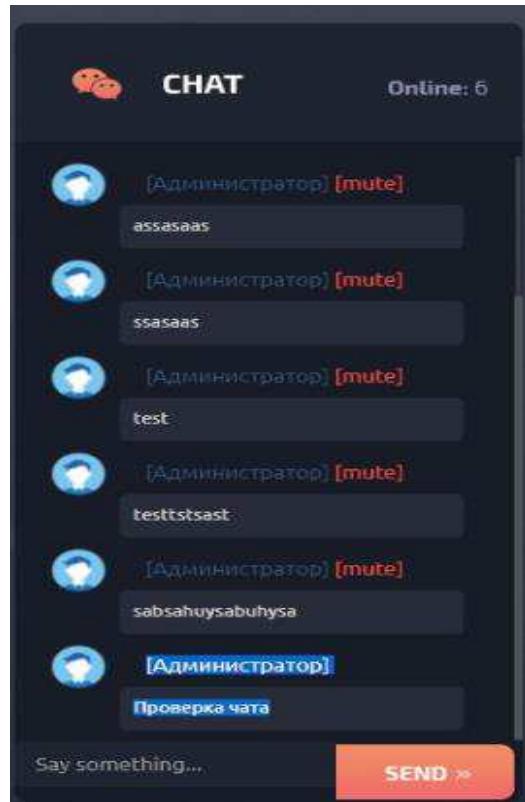


Рисунок 35 — Чат

## **ЗАКЛЮЧЕНИЕ**

В результате бакалаврской работы был создан веб-портал, позволяющий людям минимизировать инвестиционные затраты на приобретение внутриигровых предметов.

Для достижения поставленной цели были решены следующие задачи:

- изучить предметную область;
- рассмотреть существующие аналоги;
- разработать веб-портал.

В результате действий, описанных в первой главе, был проведен анализ предметной области, что позволило выявить требования к разрабатываемой информационной системе. В частности, для решения данной задачи была построена диаграмма вариантов использования, что позволило избежать несоответствий и удостовериться, что предметная область понята правильно.

Результатом второй главы стало решение задачи проектирования веб-портала. В рамках решения данной задачи были построены SADT-диаграммы, спроектирована база данных и выбраны инструменты для реализации веб-портала.

При разработке было выявлено что, выбор использованных технологий полностью себя оправдал и позволил разработать полноценный веб-портал для достижения поставленной цели.

Разработка велась на платформе Laravel Framework на языке программирования PHP с использованием HTML, CSS, JS, база данных MySQL, веб-сервера Apache.

В дальнейшем планируется внедрение web-портала и по необходимости добавление новых функций.

## **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Андерсон, С. Приманка для пользователей : создаем привлекательный сайт / С. Андерсон. – Москва : Питер, 2013. – 234 с.
2. Бейли, Л. Изучаем SQL. / Л. Бейли – Санкт-Петербург: Питер, 2012. – 592 с.
3. Введение в JavaScript [Электронный ресурс] : особенности и уникальность. / Современный учебник Javascript. – Режим доступа: <https://learn.javascript.ru/intro>
4. Вин, Ч. Как спроектировать современный сайт : профессиональный веб-дизайн на основе сетки / Ч. Вин. – Москва : Питер, 2011. – 192 с.
5. Диаграмма прецедентов [Электронный ресурс] : / – Режим доступа: <https://ru.wikipedia.org/wiki/>
6. Зандстра, М. PHP. Объекты, шаблоны и методики программирования / Мэт Зандстра – Москва : Издательский дом «Вильямс», 2016. – 576 с.
7. Кузнецов, М. В. PHP. Практика создания Web-сайтов / М. В. Кузнецов. – Москва : БХВ-Петербург, 2016. – 894 с.
8. Крупнейший в Европе ресурс для IT-специалистов [Электронный ресурс] : Реализация MVC паттерна на примере создания сайта-визитки на PHP // «Хабрахабр». – Режим доступа: <https://habrahabr.ru/post/150267/>
9. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript / Р. Никсон. – Санкт-Петербург : Питер, 2013. – 496 с.
10. Никсон, Р. Learning PHP, MySQL, JavaScript, CSS & HTML5 : A Step-by-Step Guide to Creating Dynamic Websites / Р. Никсон. – Москва : Питер, 2016. – С. 180–197.
11. Олифер, В. Г. Компьютерные сети. Принципы, технологии, протоколы : учебник для вузов / В. Г. Олифер, Н. А. Олифер. – Санкт-Петербург : Питер, 2010. – 944 с.
12. Олищук, А. Разработка Web-приложений на PHP / А. Олищук. – Москва : Вильямс, 2006. – 352 с.

13. Пирогов, В. Ю. Информационные системы и базы данных: организация и проектирование / В. Ю. Пирогов. – Санкт-Петербург : БХВ-Петербург, 2009. – 87 с.
14. Полонская, Е. Л. Язык HTML. Самоучитель. / Е. Л. Полонская – Москва : Издательский дом «Вильяме», 2005. – 320 с.
15. Попов, В. К. Практикум по Интернет-технологиям / В. К. Попов. – Санкт-Петербург: Питер, 2002. – 162 с.
16. Реляционная база данных [Электронный ресурс] : / – Режим доступа: <https://ru.bmstu.wiki/>
17. Рязанцева, Л. Что нам стоит сайт построить / Л. Рязанцева // Библиополе. – 2008. – № 8. – С. 20–21.
18. СТО 4.2–07–2014 Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности. – Введ. 30.12.2013. – Красноярск : СФУ, 2014. – 60 с.
19. Строительство Web-сайтов / В. А. Фридман, А. В. Александров, Г. Г. Сергеев, С. П. Костин. – Москва : Триумф, 2011. – 288 с.
20. Титоров, Д. Ю. Технология создания интерактивных сайтов / Д. Ю. Титоров // Информатика : [газ. Изд. дома "Первое сентября"]. – 2010. – № 3 (февр.). – С. 13–18.
21. Хассей, Т. WordPress. Создание сайтов для начинающих / Т. Хассей. – Москва : Эксмо, 2012. – 432 с.
22. Чебыкин, Р. И. Разработка и оформление текстового содержания сайтов / Р. И. Чебыкин. – Москва : БХВ–Петербург, 2014. – 528 с.
23. Чои, В. Как спроектировать современный сайт / Вин Чои – Санкт-Петербург : Питер, 2011. – 192 с.
24. Bootstrap по-русски [Электронный ресурс] : описание фреймворка, используемые технологии. – Режим доступа: <http://mybootstrap.ru/>
25. SEO — искусство раскрутки сайтов / Э. Энж [и др.]. – Москва : БХВ-Петербург, 2011. – 592 с.

Федеральное государственное автономное  
образовательное учреждение  
высшего образования  
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и Информационных Технологий  
институт  
Информационные системы  
кафедра

УТВЕРЖДАЮ  
Заведующий кафедрой ИС  
  
подпись      Л.С. Троценко  
инициалы, фамилия  
«13» июня 2018 г.

## БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии

Минимизация инвестиционных затрат на внутриигровые предметы  
средствами веб-портала

Руководитель

  
13.06.18  
подпись, дата

С.А. Виденин

инициалы, фамилия

Выпускник

  
13.06.18  
подпись, дата

М.А. Бобин

инициалы, фамилия

Нормоконтролер

  
13.06.18  
подпись, дата

Ю.В. Шмагрис

инициалы, фамилия

Красноярск 2018