Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
институт
Информационных систем
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ П.П.Дьячук
подпись инициалы, фамилия
«____» _____ 2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 – «Информационные системы и технологии»

Интеграция децентрализованной системы в процесс аренды жилья

Руководитель	_____	ст. преподаватель	Ю.В.Шмагрис
	подпись, дата	должность, ученая степень	
Консультант	_____	к.т.н., доцент	И.А.Легалов
	подпись, дата	должность, ученая степень	
Выпускник	_____		Е.К.Кузьмин
	подпись, дата		
Нормоконтролер	_____	ст. преподаватель	Ю.В.Шмагрис
	подпись, дата	должность, ученая степень	

Красноярск 2019

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
институт
Информационных систем
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ П.П.Дьячук
подпись инициалы, фамилия
«____» _____ 2019 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

в форме бакалаврской работы

Студенту Кузьмину Егору Константиновичу

Группа: КИ15-13Б Направление: 09.03.02 «Информационные системы и технологии»

Тема выпускной квалификационной работы: «Интеграция децентрализованной системы в процесс аренды жилья»

Утверждена приказом по университету № 7237/с от 24.05.2019 г.

Руководитель ВКР: Ю.В.Шмагрис, старший преподаватель кафедры «Информационные системы» ИКИТ СФУ.

Консультант ВКР: И.А.Легалов, кандидат технических наук, доцент кафедры «Информационные системы» ИКИТ СФУ.

Исходные данные для ВКР: список требований к разрабатываемой системе, методические указания научного руководителя.

Перечень разделов ВКР: введение, общие сведения, средства разработки, разработка и реализация веб-сервиса, заключение, список использованных источников.

Перечень графического материала: презентация, выполненная в Microsoft Office PowerPoint 2010.

Руководитель ВКР _____ Ю.В.Шмагрис
подпись

Консультант ВКР _____ И.А.Легалов
подпись

Задание принял к исполнению _____ Е.К.Кузьмин
подпись

« ___ » _____ 2019 г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Интеграция децентрализованной системы в процесс аренды жилья» содержит 43 страницы текстового документа, 31 рисунок, 16 использованных источников.

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ИНФОРМАЦИОННЫЕ СИСТЕМЫ, ВЕБ-СЕРВИС, ПРОЕКТИРОВАНИЕ.

Цель работы – повышение безопасности сделок по аренде жилья, посредством создания децентрализованного приложения.

Основные задачи:

1. Провести анализ предметной области технологии блокчейн
2. Создать план разработки децентрализованного приложения
3. Разработать приложение и провести его тестирование
4. Проанализировать полученные результаты

В ходе данной работы было разработано децентрализованное приложение, которое повысило безопасность сделок по аренде жилья, посредством прозрачности и открытости децентрализованных систем.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Централизованные системы формировались тысячелетиями, перенимая из прошлого они стали основой для организации структуры взаимоотношений людей и объектов, которые их окружают. Мы привыкли к тому, что для совершения каких-либо операций требуется обойти разное количество “центров-узлов” для того, чтобы получить конечный результат. Например, перед нами встаёт какая-то проблема, покупка машины или жилья, поход в больницу, заверение у нотариуса, всё это требует прохождения нескольких учреждений, для решения нашей проблемы.

Создание и последующее развитие интернета постепенно привнесло упрощение данных процедур. Теперь покупка машины стала легче, прямо из дома можно найти сайт продажи автомобилей и все что нам останется, это договориться о покупке машины с хозяином. Но централизация никуда не делась, даже если у нас есть интернет и какая-то автоматизированная система покупки машин, нам всё равно придется идти в банк, встречаться с хозяином машины, проверять машину и др. Поэтому в конце 20-ого века были придуманы, а сейчас активно развиваются децентрализованные системы, которые позволяют распределять обязанности всем участникам процесса, представьте, что покупка машины это на сто процентов защищенная, сделка, а все условия и узлы становятся равными внутри сети. Одним действием происходят все процедуры и вам не нужно идти во все заведения, чтобы завершить сделку. Решаются такие проблемы, как погрешность и ошибка человеческого фактора, достоверность и целостность информации внутри системы.

Децентрализованные системы, а в дальнейшем и децентрализованные приложения, очень современная технология и пока что нельзя говорит, о её стопроцентном применении во всём, потребуется проверка временем. Но для развития нужны новые проекты и амбициозные решения, которые позволят

подтвердить, что это действительно нужная и более правильная вещь чем централизованные системы.

В данной бакалаврской работе будет произведена интеграция децентрализованной системы в процесс аренды жилья.

Целью данной работы будет повышение безопасности сделок по аренде жилья, посредством создания децентрализованного приложения.

Можно сформулировать задачи:

1. Провести анализ предметной области технологии блокчейн
2. Создать план разработки децентрализованного приложения
3. Разработать приложение и провести его тестирование
4. Проанализировать полученные результаты

Глава 1 Анализ предметной области

1.1 Децентрализация

Начиная говорить про блокчейн, разбирать его нужно как прикладную технологию, которая может применятся в разных сферах жизнедеятельности человека связанных с информационными технологиями. Но в данный момент больше всего блокчейн организован на проведение финансовых операций. Как технология, он позволяет сделать информацию более безопасной и конфиденциальной благодаря цепочке блоков, которая хранит в себе данные о транзакциях всех лиц в сети.

1.1.1 Основы построения Блокчейна

Истоки блокчейна идут из создания криптовалюты Bitcoin. Программист(ы) Сатоши Накамото, решил, что будущее финансовых отношений лежит, через криптовалюты (речь о которых пойдет в следующих пунктах), электронной валюты, которая работает благодаря криптографии и распределенных баз данных. Распределенные базы данных это такие базы данных, которые распределены в сети между её участниками. Например, если в сети 3 человека, то центра откуда будут браться информация нет, у каждого участника будет своя реплицированная база данных. Данный пункт является одним из основных постулатов криптовалюты.

Что из себя представляет реплицированная база данных. Это прозрачное ведение и учет информации. Репликация и распределенность основная концепция реплицированных баз данных.

Данный метод построения баз данных заимствовал и блокчейн.

1.1.2 Цепочка блоков

Цепочка блоков полностью составляют структуру блокчейна. Это база транзакций, подкрепленная криптографией, в особенности хэш-суммами. Хэш-сумма – это результат хэширования, когда массив информации проходит, через определенный алгоритм и на выходе получается набор символов. Стоит заметить, что набор символов для одинаковой информации должен быть всегда одинаковый, так как смысл хеширования теряется.

База транзакций обрабатывается каждым участником сети, то есть происходит распределенная обработка. Цепочка содержит в себе все транзакции, произведенные в системе.

Например, у нас есть блокнот, доступ к которому есть у нескольких человек, каждый день в нём производятся изменения, эти изменения и будут транзакциями. Каждый, кто хочет получить доступ к цепочке блоков данного блокнота увидит, кто и когда изменил информацию, и какая-информация была изменена. Это возможно благодаря хэш-суммам, необходимым для целостности информации. Каждая транзакция будет формировать блоки целостности, для того чтобы подтвердить информацию. Две записи в блокноте образуют две транзакции, далее они формируются в один блок, содержащий хэш-сумму прошлых транзакций, где подтверждаются их правильность, посредством сравнения, если будет найдено отличие, то данный блок не будет подтвержден. Более подробную картину цепочки блоков можно увидеть на рисунке 1.

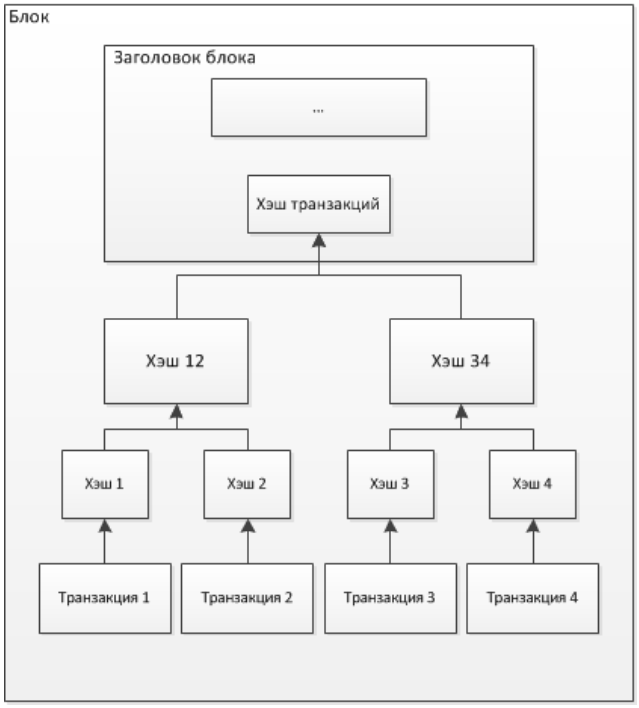

Рисунок 1 – Структура цепочки блоков

Таким образом, любая криптовалюта содержит миллионы таких цепочек, они распределено хранятся на миллионах компьютерах пользователей по всему миру.

1.2 Основы криптографии криптовалют

Безопасность в интернет технологиях превыше всего, защита системы, конфиденциальность данных является одной из основ доверия к системе. Криптовалюты используют криптографию для защиты своей системы, она основана исключительно на математике, поэтому любой незнакомый участник сети, может доверять другому участнику сети и в этом случае инструмент доверия – это математические функции, шифры и цифровые подписи.

Криптография - это метод использования передовых математических принципов для хранения и передачи данных в определенной форме, так что только те люди, для которых они предназначены, могут читать и обрабатывать информацию. Криптография используется уже тысячи лет для тайного общения. Самое раннее использование криптографии было зафиксировано в

гробнице, найденной в Древнем царстве в Египте около 1900 года до нашей эры.

1.2.1 Виды шифрования

Наиболее важным инструментом криптографии является шифрование. Шифрование в обычном понимании — это возможность в целях безопасности передавать информацию в виде бессмысленного набора символов, который в дальнейшем можно привести к изначальной форме с помощью ключей и алгоритмов.

Проведем обзор способов шифрования.

Самым старым и известным методом шифрования является симметричная криптография. Работает по принципу наличия ключей у участников передачи информации. С помощью ключа производится шифрование и дешифрование информации. На рисунке 2 приведена структура работы такого шифрования.

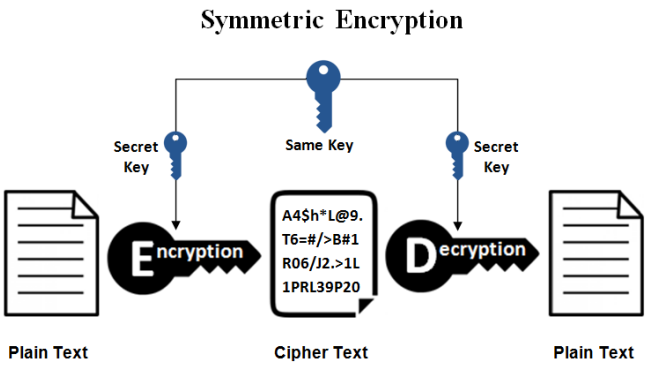

Рисунок 2 – Симметричное шифрование

Следующим способом шифрования является ассиметричное шифрование.

Данную концепцию придумал британский математик Джеймс Эллис. Он предположил, что информацию нужно шифровать одним ключом, а процесс дешифрования должен производится совершенно другим ключом, который будет отправлен вместе с сообщением.

Асимметричное шифрование использует два ключа для шифрования информации. Секретные ключи обмениваются через интернет или другую большую сеть. Важно отметить, что любой пользователь с приватным ключом может расшифровать сообщение, и поэтому асимметричное шифрование использует два связанных ключа для повышения безопасности. Открытый ключ доступен всем, кто захочет отправить вам сообщение. Второй приватный ключ хранится в секрете.

Информация, зашифрованная с использованием открытого ключа, может быть дешифрована только с использованием приватного ключа, в то время как сообщение, зашифрованное с использованием приватного ключа, может быть дешифровано с использованием открытого ключа.

Например, Вася хочет отправить сообщение Кириллу. У Кирилла есть открытый ключ и закрытый ключ, он отправляет открытый ключ Васе, а Вася, получив данный ключ вставляет его в передаваемое сообщение и только Кирилл сможет его расшифровать.

Математическим языком, это можно назвать односторонней функцией. A — открытый ключ, a — приватный. Ключи математически связаны друг с другом через функцию:

$$A = f(a) \tag{1.1}$$

На рисунке 3 приведена структура работы ассиметричного шифрования.

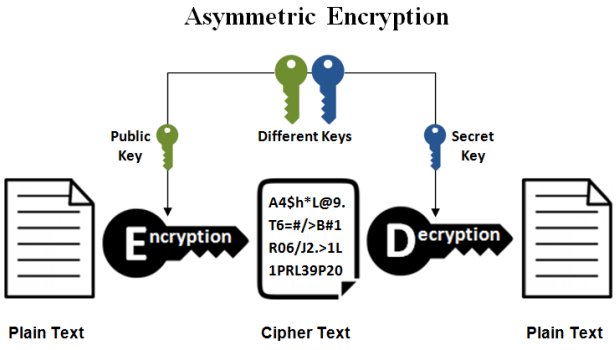

Рисунок 3 – Асимметричное шифрование

1.2.2 Криптография с помощью эллиптических кривых

Bitcoin и другие криптовалюты используют в своей криптографии шифрование с помощью эллиптических кривых.

Эллиптическая кривая — это внешне довольно простая функция, как правило, записываемая в виде так называемой формы Вейерштрасса:

$$y^2 = x^3 + ax + b \tag{1.2}$$

В криптовалютах используется кривая под названием SECP256K1. Уравнение для кривой SECP256K1 имеет вид:

$$y^2 = x^3 + 7 \tag{1.3}$$

На рисунке 4 показан вид данной кривой.

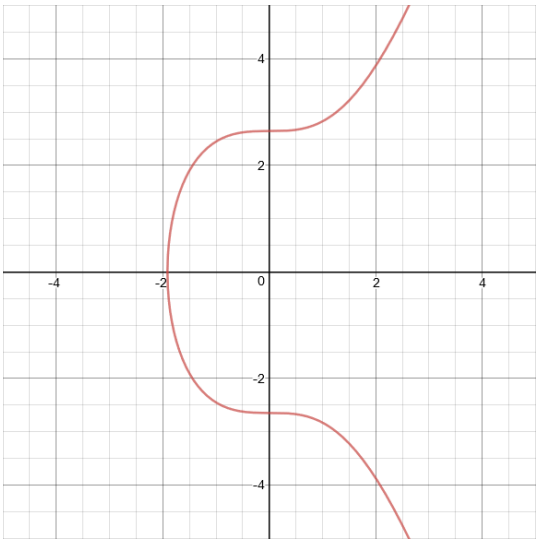

Рисунок 4 – Кривая SECP256K1

Эллиптические кривые имеют несколько свойств:

1) Невертикальная линия, пересекающая две некасательные точки на кривой, пересечет третью точку на кривой.

2) Суммой двух точек на кривой P + Q называется точка R, которая является отражением точки -R (построенной путем продолжения прямой (P; Q) до пересечения с кривой) относительно оси X.

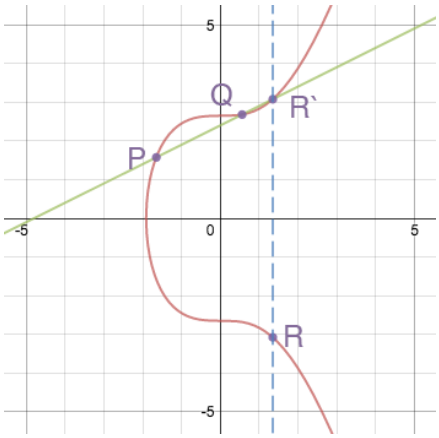

Рисунок 5 – Свойства эллиптической кривой

Если же провести прямую через две точки, имеющие координаты вида P (a, b) и Q (a, -b), то она будет параллельна оси ординат. В этом случае не будет третьей точки пересечения. Чтобы решить эту проблему, вводится так называемая точка на бесконечности (point of infinity), обозначаемая как O. Поэтому, если пересечение отсутствует, уравнение принимает следующий вид:

$$P + Q = O \tag{1.4}$$

В эллиптической криптографии (ECC) используется такая же кривая, только рассматриваемая над некоторым конечным полем. Конечное поле в контексте ECC можно представить, как предопределенный набор положительных чисел, в котором должен оказываться результат каждого вычисления.

Например, 9 mod 7 = 2. Здесь мы имеем конечное поле от 0 до 6, и все операции по модулю 7, над каким бы числом они ни осуществлялись, дадут результат, попадающий в этот диапазон.

Эллиптическая кривая биткойна, определенная на конечном поле по модулю 67, изображена на рисунке 6.

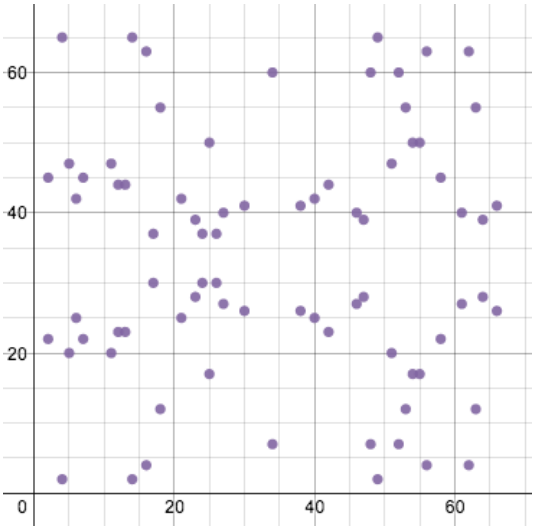

Рисунок 6 – Эллиптическая кривая биткойна, определенная на конечном поле по модулю 67

В протоколе биткойна зафиксирован набор параметров для эллиптической кривой и её конечного поля, чтобы каждый пользователь использовал строго определенный набор уравнений. Среди зафиксированных параметров выделяют уравнение кривой (equation), значение модуля поля (prime modulo), базовую точку на кривой (base point) и порядок базовой точки (order). Этот параметр подбирается специально и является очень большим простым числом.

В биткойне кривая secp256k1 используется совместно с алгоритмом цифровой подписи ECDSA (elliptic curve digital signature algorithm). В ECDSA секретный ключ — это случайное число между единицей и значением порядка. Открытый ключ формируется на основании секретного: последний умножается на значение базовой точки. Уравнение имеет следующий вид:

$$\text{Открытый ключ} = \text{секретный ключ} * \text{базовая точка} \quad (1.5)$$

Вычисление открытого ключа выполняется с помощью тех же операций удвоения и сложения точек. Это тривиальная задача, которую обычный персональный компьютер или смартфон решает за миллисекунды. А вот обратная задача (получение секретного ключа по публичному) — является проблемой дискретного логарифмирования, которая считается вычислительно сложной (хотя строгого доказательства этому факту нет).

Когда пара секретный/публичный ключ получена, её можно использовать для подписи данных. Эти данные могут быть любой длины. алгоритм подписи данных z выглядит следующим образом:

Здесь, G — базовая точка, n — порядок, d — приватный ключ, Q – публичный ключ, k – автогенерирующееся число.

1) Открытый ключ

$$Q = dG \tag{1.6}$$

2) Умножить G на случайное число «k» и построить эту точку на графике, координатами этой точки являются (x, y). т. е.:

$$(\mathrm{x}, \mathrm{y}) = kG \tag{1.7}$$

3) Затем определяется два значения r и s такие, что:

$$r = x \bmod n \tag{1.8}$$

$$s = (z + rd)\, k^{-1} \bmod n \tag{1.9}$$

4) Генерация точки (r, s), отправка на подпись.

1.2.3 Электронная подпись и проверка в Bitcoin

Электронная подпись является преобразованным документом в результате криптографических алгоритмов. Она позволяет проверить целостность информации в электронном виде, авторство и подтвердить факт подписания информации. Построена на основе публичных и приватных ключей.

Создание электронной подписи работает следующим образом. Отправитель хочет подписать документ. Первым делом выполняется хеширование документа, к хэш-сумме добавляется приватный ключ отправителя, а дальше полученный набор данных уходит на подтверждение отправителю. Получатель, используя публичный ключ отправителя дешифрует документ и получает хэш-сумму документа, где происходит его дальнейшее сравнение и подтверждение.

Bitcoin предусматривает похожий сценарий электронной подписи о совершенных транзакциях. С помощью функции validate_signature встроенной в Bitcoin будет происходить верификация, что отправитель отправил деньги получателю. А остальные участники сети, могут проверить, что данная транзакция и прошла, используя открытый ключ отправителя.

1.3 Транзакции в Bitcoin

Транзакция - это подписанные данные, которые образуют блокчейн криптовалюты. В ней хранятся такие параметры как: количество валюты (amount) соответствующее определенному публичному ключу – адресу пользователя Bitcoin, а также количество отправляемой валюты, ссылки на предыдущие транзакции, принадлежащие этому ключу, входы (input) – записи адреса отправителя, выходы (output) – записи адреса получателя. Общий вид транзакции приведен на рисунке 7.

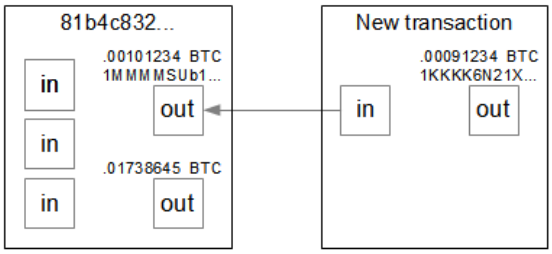

Рисунок 7 – Структура транзакции

Непосредственно сам процесс: Вася хочет отправить Кириллу определенное количество Биткоинов. С помощью приватного ключа будет создана заявка на перевод. Транзакция Васи будет включать в себя публичный код Васи (вход), публичный код Кирилла (выход), количество валюты (amount). Далее произойдет процесс подтверждения транзакции по алгоритму криптографии приведенным ранее.

Процесс формирования и подтверждения блоков транзакций может быть затянут, так как криптовалюты работают за счёт вычислительных способностей компьютеров майнеров – пользователей, который находят блоки. Скорость подтверждения транзакций различная в 2018 году для Bitcoin подтверждение составляло от 10 до 60 минут, для криптовалюты Ethereum 15 - 30 секунд.

1.4 Децентрализованные приложения

В 2015 году была запущена криптовалюта Ethereum, разработанная русско-канадским программистом Виталием Бутериным. Особенностью Ethereum являлось создание смарт-контрактов. Смарт-контракт – это компьютерный алгоритм, который выполняется внутри децентрализованной системы и с помощью условий, заданных в коде, контролирует валюты между участниками контракта. Главным условием создания является возможность описать процессы математическим языком.

Смарт-контракт содержит в себе: подписантов, предмет сделки, условия сделки, связь с децентрализованной платформой

Использование смарт-контрактов в программировании позволило создавать веб-приложения или же программы, которые используют возможности криптовалют. Такие продукты стали называться децентрализованными приложениями. Пример такого приложения приведен на рисунке 8. Это русская платформа для размещения статей, которые могут быть оплачены в зависимости от её качества и заинтересованности аудитории, все действия выполняются с помощью блокчейна и смарт-контрактов Ethereum.

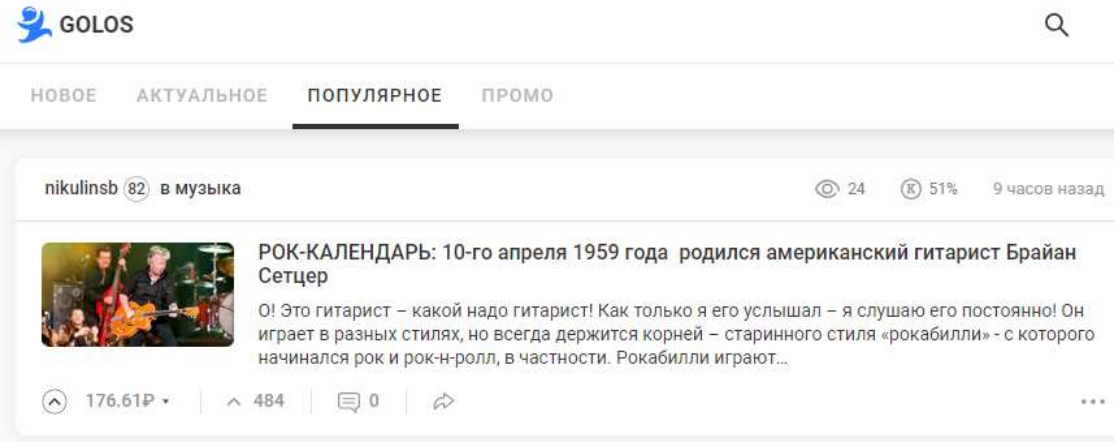

Рисунок 8 - Пример децентрализованного приложения, сайт Golos.io

1.5 Подведение итогов первой главы

В первой главе бы проведен анализ предметной области децентрализованных систем и основ криптографии криптовалют. Были выявлены особенности криптовалюты Bitcoin, принципы ее работы. Затронуты возможности децентрализованных приложений и их примеры.

Полученная информация помогла в принятии решения о методологии разработки приложения, которое интегрирует децентрализованную систему в процесс аренды жилья.

Глава 2 Средства и методы разработки

2.1 Описание и функционал проектируемой системы

Информационная система направлена на проведение сделок между арендодателями и людьми, которые хотят арендовать недвижимость. Помимо автоматизации проведения данных сделок, также существует особенность данной информационной системы, которая заключается в применении децентрализации и приведению её к типу децентрализованного приложения посредством смарт-контрактов криптовалюты Ethereum.

Внедрение децентрализованной системы позволит повысить безопасность сделок. Безопасность будет заключаться в контроле сделки с помощью смарт-контракта.

Функционал у данной системы можно разделить на два блока, первый блок будет отвечать за взаимодействие пользователя с системой, второй же блок направлен непосредственно на функционал смарт-контрактов.

Первый блок:

1) Регистрация и авторизация пользователей – возможность пользователей регистрироваться и с помощью авторизации получать права на последующий функционал

2) Создание записей об аренде недвижимости – возможность пользователей создать запись о сдаче/снятии недвижимости исходя из целей пользователя.

3) Редактирование записей пользователей – возможность пользователей редактировать записи, добавлять новую информацию, либо удалять информацию вплоть до удаления записи.

4) Фильтрация записей – возможность пользователей фильтровать записи по локации, стоимости и других параметров.

Второй блок:

1) Создание сделки – генерация смарт-контракта при решении пользователя сдать/снять недвижимость

2) Оплата сделки – оплата пользователем сделки в сети Ethereum

3) Расторжение сделки – расторжение сделки в случае выявленных проблем с помощью смарт-контракта

2.2 Требования к системе

После проведенного анализа предметной области и рассмотрения будущих функциональных блоков, были выдвинуты следующие требования к разрабатываемой системе:

1) Открытость (возможность наблюдать за процессами без ограничений)
2) Надежность (платформа стабильно работает в течении длительного времени)
3) Удобство разработки и наличие документации
4) Отказоустойчивость
5) Понятность системы для пользователя

2.3 Выбор архитектуры ИС

Выбор архитектуры разрабатываемой системы влияет на её дальнейшую жизнь, поэтому важно выбирать наиболее подходящий вид построения и развертывания ИС.

Существует несколько видов архитектур ИС:

1) Одноуровневая, когда все выполняется непосредственно на одном компьютере и в одном месте, не существует такого понятия как база данных, есть лишь программный код, который предоставляет информацию пользователю. Например, стандартное приложение Windows «Блокнот»

2) Двухуровневая, все также выполняется на одном компьютере, но в архитектуру включается база данных, из которой с помощью запросов можно выводить информацию.

3) Многоуровневая архитектура, ставшая популярной после создания интернета, когда каркас представляет из себя клиент-серверное приложение. Пользователь (клиент) делая какие-либо действия передаёт на сервер запросы, которые впоследствии обрабатываются и посылаются обратно пользователю. Сейчас данной архитектуры придерживается практически все разработчики веб-приложений.

Для разработки децентрализованного приложения аренды жилья будет использоваться клиент-серверная архитектура с подключением к сети криптовалют Ethereum.

2.4 Средства разработки и проектирования ИС

Проектирование системы будет производится с помощью CASE-средств, в частности диаграмм UML, так как они позволяют в упрощенном виде показывать отношения между частями системы. Программным средством для создания диаграмм будет StarUML и онлайн сервис draw.io

Серверная часть будет реализована на языке программирования Python, с использованием фреймворка Django, который позволяет использовать паттерн программирования MVC.

Для взаимодействия с децентрализованным узлом криптовалюты Ethereum будет использоваться интерфейс для языка программирования Python – web3.py, описывающий основные методы взаимодействия со смарт-контрактами и сетью Ethereum.

Смарт-контракт, реализующий основные процессы оплаты и заключения договора аренды, будет написан на языке программирования Solidity.

СУБД используемое в системе SQlite3 – простая и понятная база данных, для небольших проектов.

Сервер баз данных не будет использоваться, на это есть несколько причин:

1) Малое количество данных для хранения
2) Прототип, не имеющий цели выходить на рынок
3) Часть данных будет распределена в сети Ethereum

Основная часть разработки системы будет производится в среде разработки PyCharm компании JetBrains.

Компонентную диаграмму архитектуры системы можно увидеть на рисунке 9.

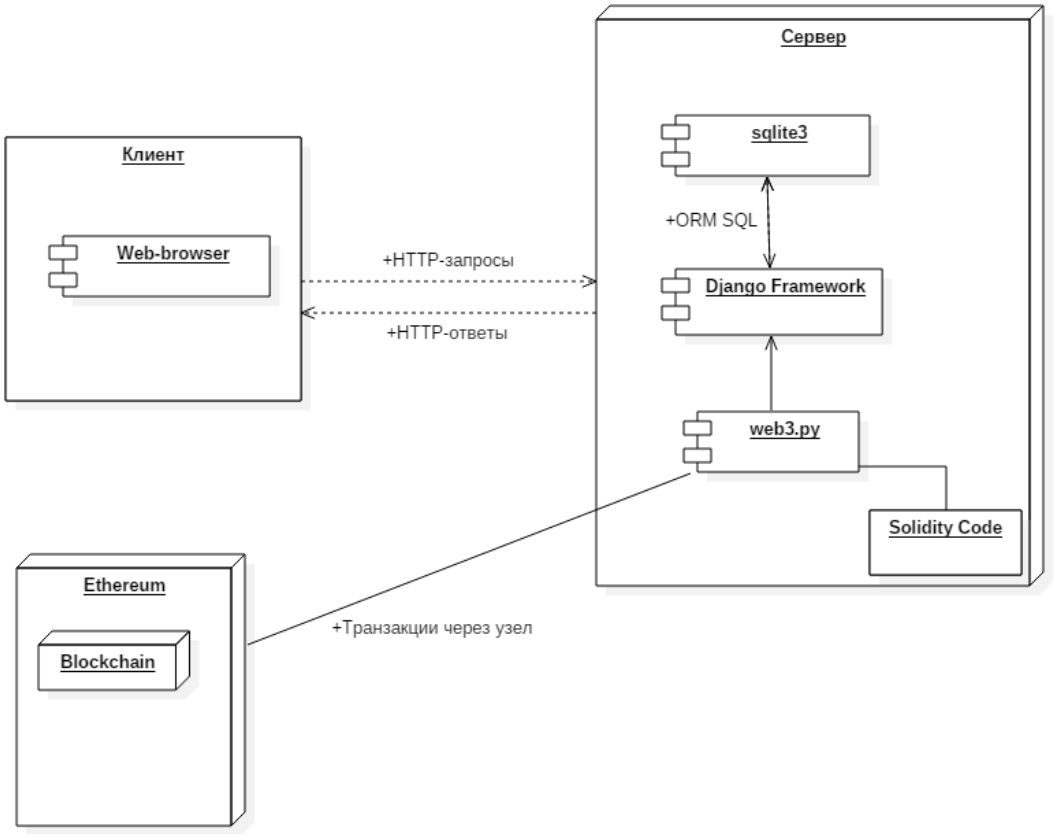

Рисунок 9 – Архитектура системы на диаграмме компонентов UML

2.5 Смарт-контракты

Для соединения первого функционального блока системы с децентрализованной системой, будет использованы смарт-контракты. Они позволяют прописать логику и структуру выполнения действий между

участниками сделки. Смарт-контракты используют статически типизированный JavaScript-подобный язык программирования, созданный для разработки самовыполняющихся контрактов, исполняющихся на виртуальной машине Ethereum (EVM). Основой методологии программирования являются структуры данных. Процесс компиляции переводит код в байт-код, к которому можно обращаться посредством двоичного интерфейса приложений ABI.

Основными объектами внутри смарт-контрактов являются адрес пользователя в сети Ethereum, адрес самого смарт-контракта и интерфейс ABI. На рисунке 10 показана структура двоичного интерфейса контракта по аренде жилья.

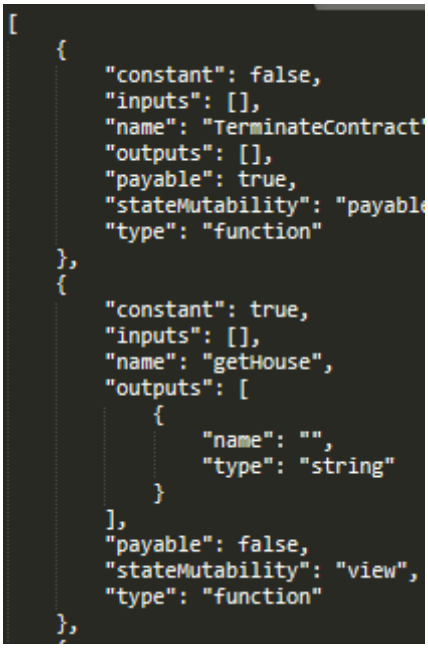

Рисунок 10 – Интерфейс ABI

Процесс разработки и компиляции смарт-контрактов будет производится в онлайн среде разработки Remix. Данная среда позволяет производит написание, компиляцию, запуск написанных контрактов, дебаггинг, тестирование и др. С помощью неё получится извлечь ABI и bytecode контракта, который понадобится в дальнейшем для написания логики интерфейса web3.py.

Для того, чтобы произведенные действия в смарт-контракте переносились в сеть Ethereum, нужно средствами web3.py произвести соединение с децентрализованным провайдером (узлом). Это может быть либо эмулятор сети, о котором будет упомянуто в процессе реализации, локальный провайдер, с помощью которого на компьютер скачивается сеть Ethereum, либо HTTP-провайдеры, расположенные удалённо. В этой работе будет использоваться удалённый провайдер Infura.io, с помощью своих методов, он позволяет отправлять данные в сеть. Сетью является тестовая Kovan Network, которая создана специально для разработчиков, так как в ней не существует настоящей валюты. На рисунке 11 показана веб-страница провайдера, со статистикой вызовов методов к сети за промежуток времени.

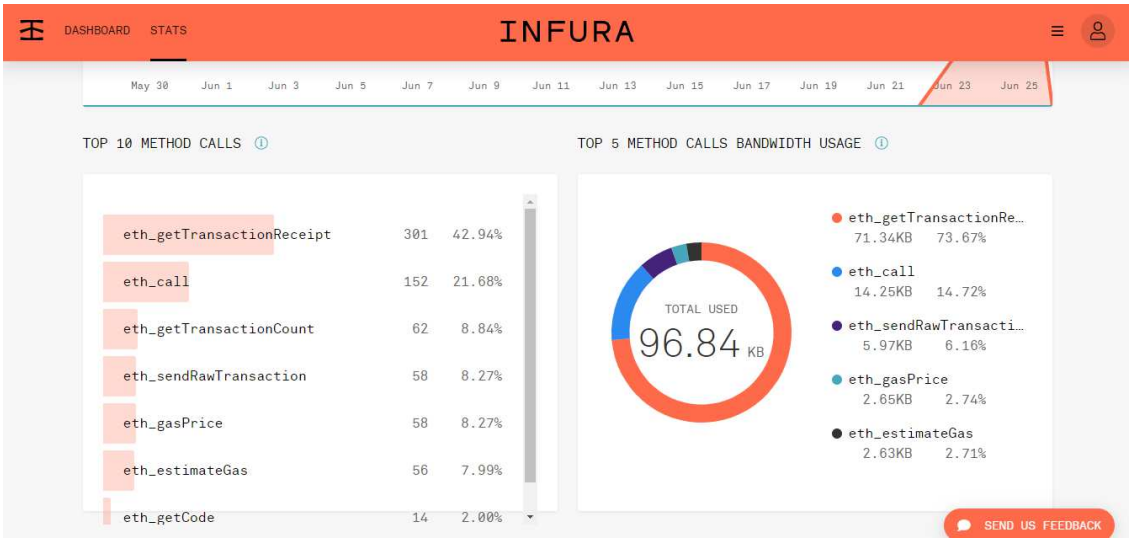

Рисунок 11 – Страница статистики Infura.io

Вызов функций смарт-контракта в сети подразумевает наличие приватных ключей, для того, чтобы существовала возможность подписывать транзакции, так как без этого не будет понимания кто производит эти действия. Для этого пришлось получить приватные ключи с помощью функций расширения для браузера Chrome - Metamask. Изначально были созданы аккаунты в тестовой сети Ethereum – Kovan Network, далее из интерфейса самого расширения были запрошены приватные ключи.

Глава 3 Реализация системы

3.1 Проектирование по средствам UML-диаграмм

Для проектирования системы была выбрана методология UML, так как она позволяет более свободно объяснить структуру связи и отношения внутри системы.

UML – унифицированный язык моделирования (Unified Modeling Language) – это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования.
Его можно использовать для визуализации, спецификации, конструирования и документирования программных систем.

Словарь UML включает три вида строительных блоков:

1) Диаграммы
2) Сущности
3) Связи

Существует несколько типов диаграмм, но для описания нашей системы будем использовать только 3 типа:

1) Диаграмма классов
2) Диаграмма деятельности
3) Диаграммы вариантов использования (Use Case)

3.1.1 Основные диаграммы UML для проектируемой системы

1) Диаграмма классов представляет собой некоторый граф, вершинами которого являются элементы типа "классификатор", которые связаны различными типами структурных отношений.

На рисунке 12 показана диаграмма классов для процесса аренды жилья. Существуют классы пользователей, сделок, объявлений и контрактов.

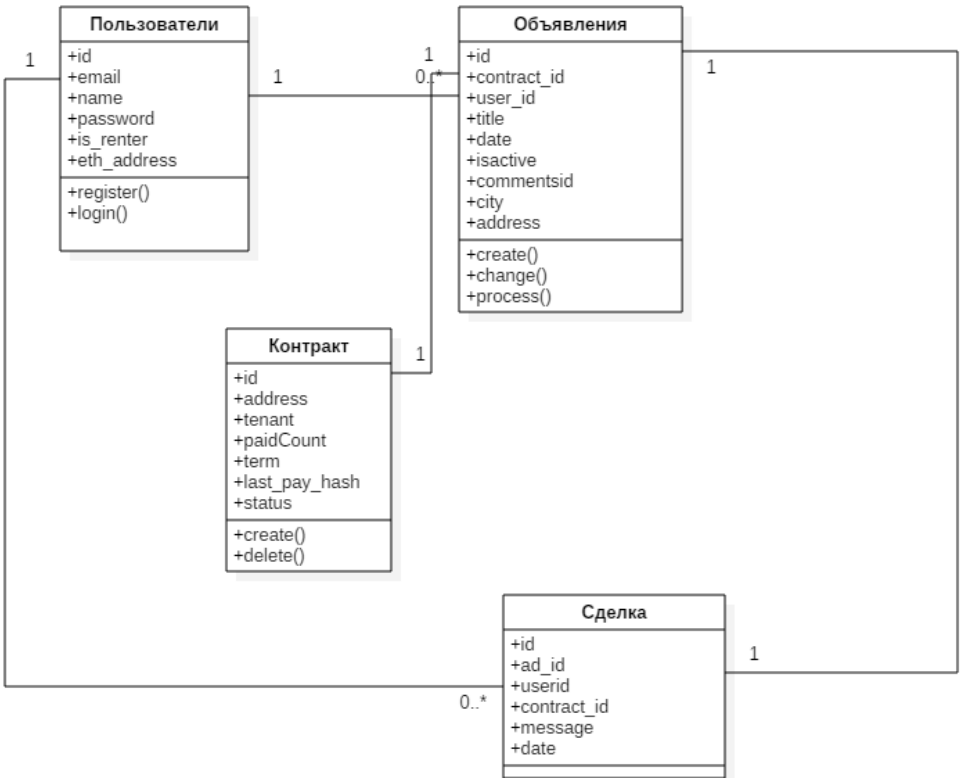

Рисунок 12 - Диаграмма классов процесса аренды жилья внутри веб-приложения

2) Диаграмма деятельности акцентирует внимание на последовательности выполнения определенных действий, которые в совокупности приводят к получению желаемого результата. Они могут быть построены для отдельного варианта использования, кооперации, метода и т. д. Диаграммы деятельности являются разновидностью диаграмм автоматов, но если на второй основное внимание уделяется статическим состояниям, то на первой – действиям.

Графически диаграмма деятельности, как и диаграмма автоматов, представляется в виде ориентированного графа, вершинами которого являются действия или деятельности, а дугами – переходы между ними. Напомним, что в UML действие – это атомарная операция, выполнение которой не может быть прервано, а деятельность – составная операция, с возможностью ее прерывания. Переход к следующему действию или деятельности срабатывает сразу по их завершении.

На рисунке 13 приведена диаграмма деятельности процесса создания сделки об аренде жилья, её подтверждения и оплаты.

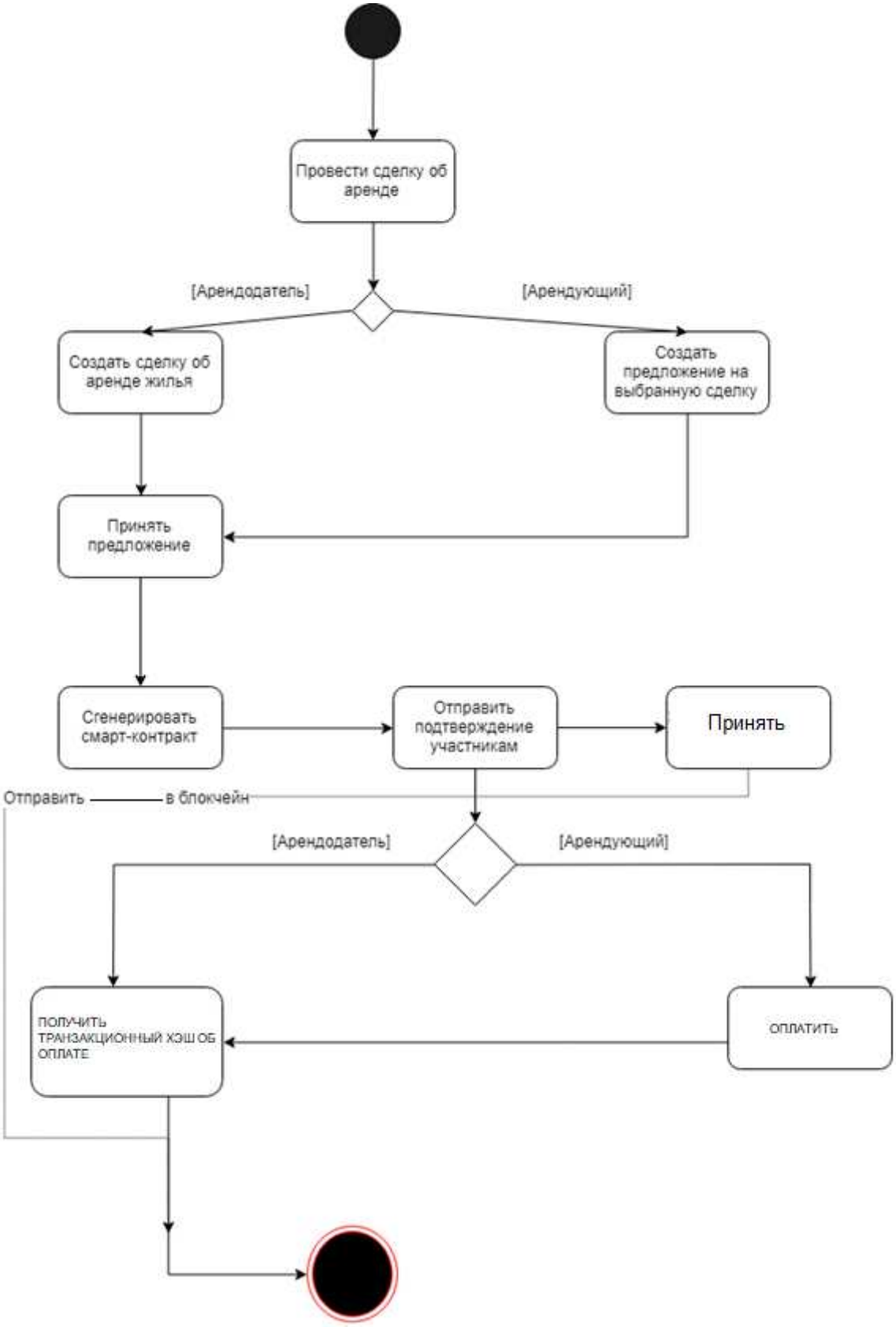

Рисунок 13 - Диаграмма деятельности процесса создания и оплаты сделки

3) Диаграмма вариантов использования (сценариев поведения, прецедентов) является исходным концептуальным представлением системы в процессе ее проектирования и разработки. Данная диаграмма состоит из актеров, вариантов использования и отношений между ними. При построении диаграммы могут использоваться также общие элементы нотации: примечания и механизмы расширения.

Суть данной диаграммы состоит в следующем: проектируемая система представляется в виде множества актеров, взаимодействующих с системой с помощью так называемых вариантов использования. При этом актером (действующим лицом, актантом, актором) называется любой объект, субъект или система, взаимодействующая с моделируемой системой извне. В свою очередь вариант использования – это спецификация сервисов (функций), которые система предоставляет актеру. На рисунке 14 приведена Use-case диаграмма сценариев поведения арендатора и арендодателя.

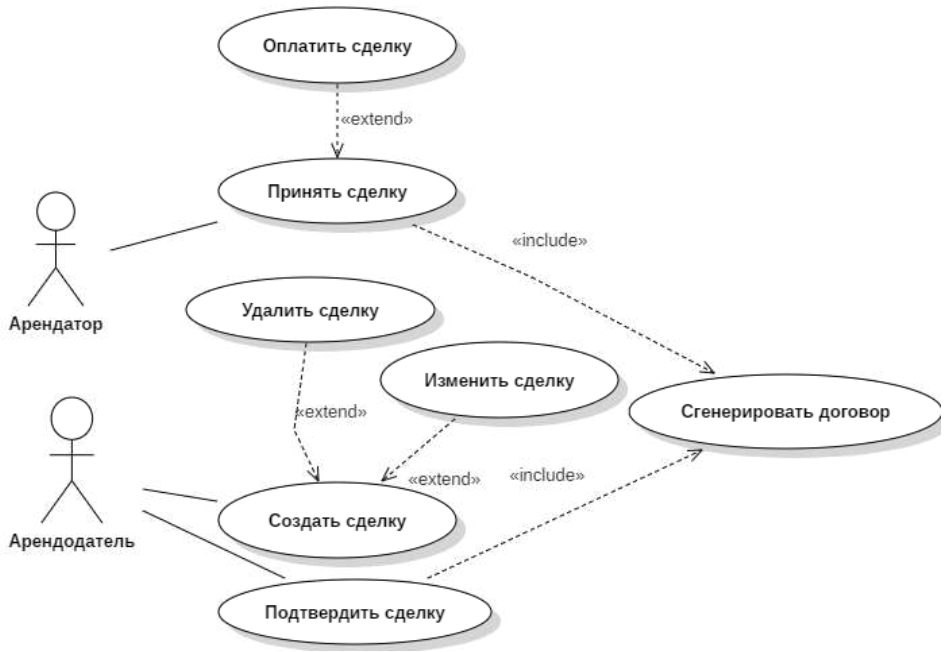

Рисунок 14 - Use-case диаграмма веб-приложения по аренде жилья

3.2 Реализация первого функционального блока

Создание блока взаимодействия пользователя с сайтом должно было быть простым и дружелюбным для пользователя. Для решения функций первого блока, был использован фреймворк языка Python для разработки веб-приложений Django. В краткие сроки удалось создать веб-приложение, удовлетворяющее требованиям системы.

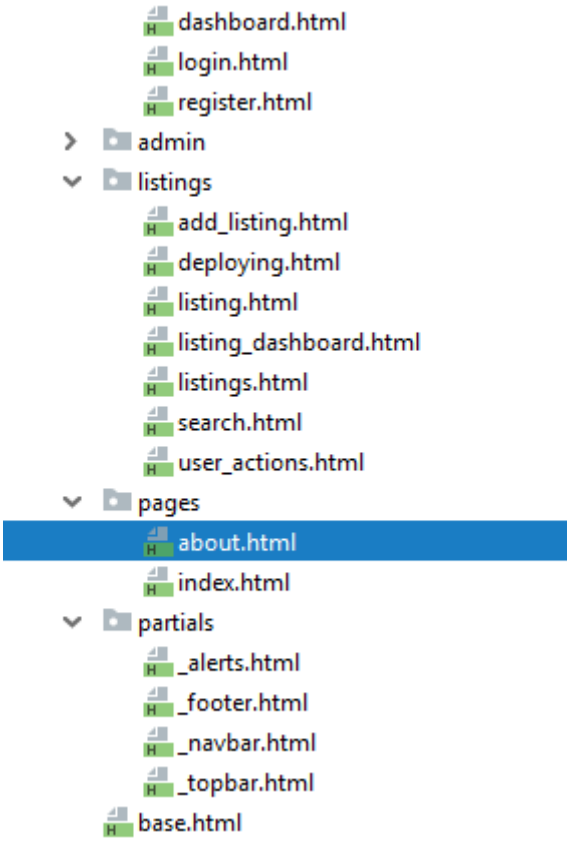

Рисунок 15 – Структура шаблонов HTML-разметки в среде разработки PyCharm

Компоненты веб-приложения разделены на отдельные структуры, в которых прописана логика их взаимодействия в целом: аккаунты, сделки, авторизация, подсистема путей, шаблоны разметки, всё это образует конечное приложение. На рисунке 15 показаны шаблоны, образующие сайт, структура шаблонов (Templates), является одной из неотъемлемых составляющих фреймворка Django.

На рисунке 16 показана главная страница разработанного веб-приложения.

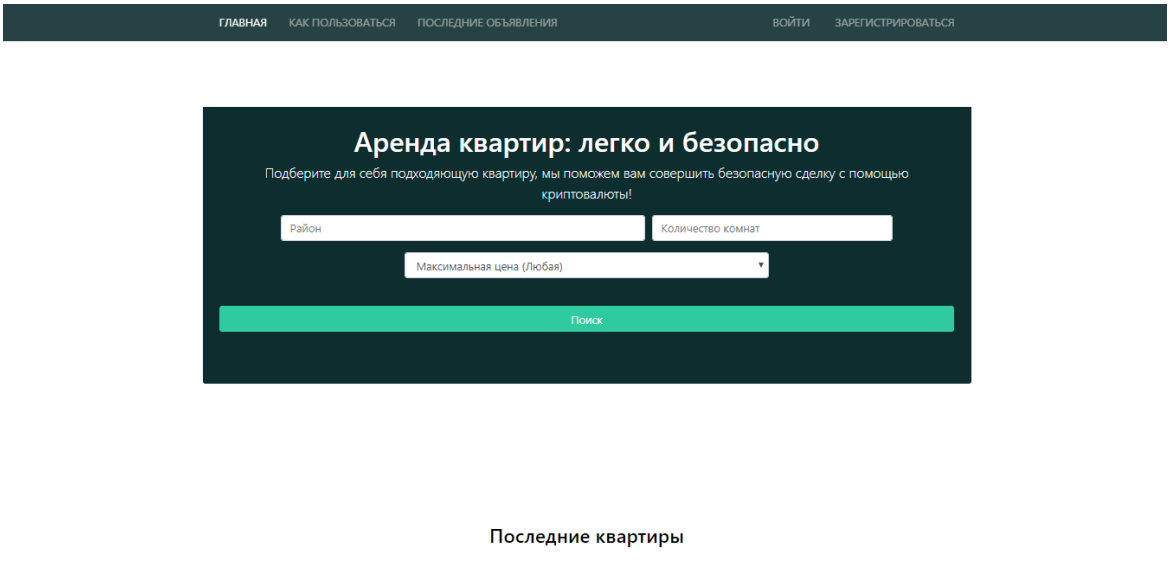

Рисунок 16 – Главная страница веб-приложения

Главная страница состоит из системы поиска по квартирам, наверху расположен навигационный бар, который помогает пользователям ориентироваться по разделам сайта

Для полного доступа на сайт, пользователь должен зарегистрироваться и пройти авторизацию на соответствующих страницах.

Существует раздел, который предоставляет пользователям небольшое руководство по веб-приложению с необходимым условием для его пользования. На рисунке 16 показан раздел с руководством.

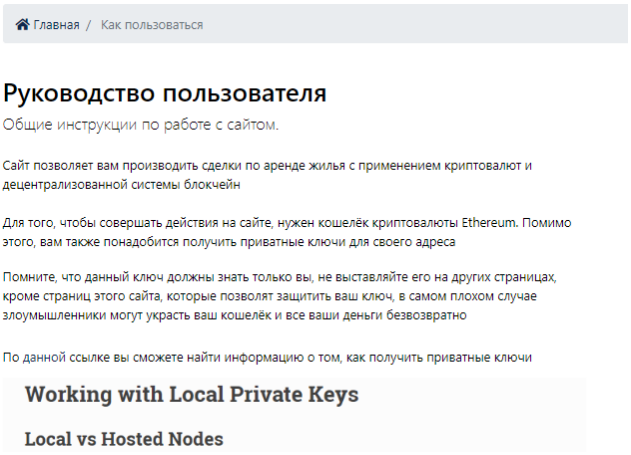

Рисунок 17 – Руководство пользования веб-приложением

Важным условием использования сайта является наличие у пользователей приватных ключей их аккаунтов в сети криптовалюты Ethereum. Приватные ключи, нужны для подписей и отправления транзакций в децентрализованный узел. На странице руководства пользователь сможет найти инструкции о том, как получить данный ключ.

После того как получены все необходимые данные пользователь может пройти регистрацию на сайте, регистрация включает в себя чекбокс, с помощью которого происходит разделение прав пользователей. В модели пользовательских аккаунтов введена булева переменная is_renter (вы арендодатель), в зависимости от её значения будет происходить различное отображение элементов на сайте. Большую заслугу в разграничение прав пользователей играют теги Django, используемые в HTML-шаблонах. На рисунке 18 можно увидеть окно регистрации пользователей.

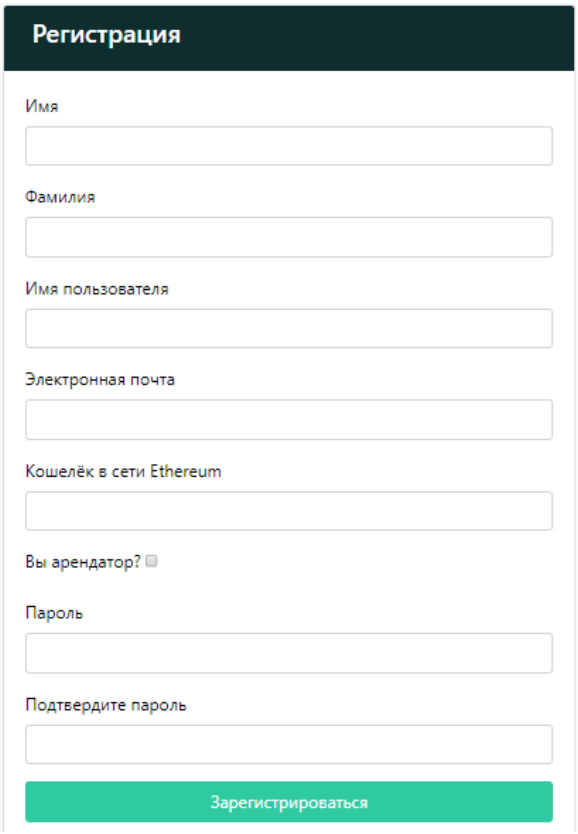

Рисунок 18 – Окно регистрации

После регистрации пользователь проходит процедуру авторизации и попадает в профиль. Как было сказано ранее в зависимости от прав пользователя будут отображаться разные элементы, но в целом вид профилей одинаковый, для арендодателя и арендующего.

Следующим шагом для арендодателя будет создание своего объявления об аренде. Создание объявлений осуществляется с помощью встроенных в Django форм – элемент, куда пользователь вводит свои данные, для того чтобы алгоритмы смогли обработать полученную информацию. На рисунке 19 показана форма добавления нового объявления.

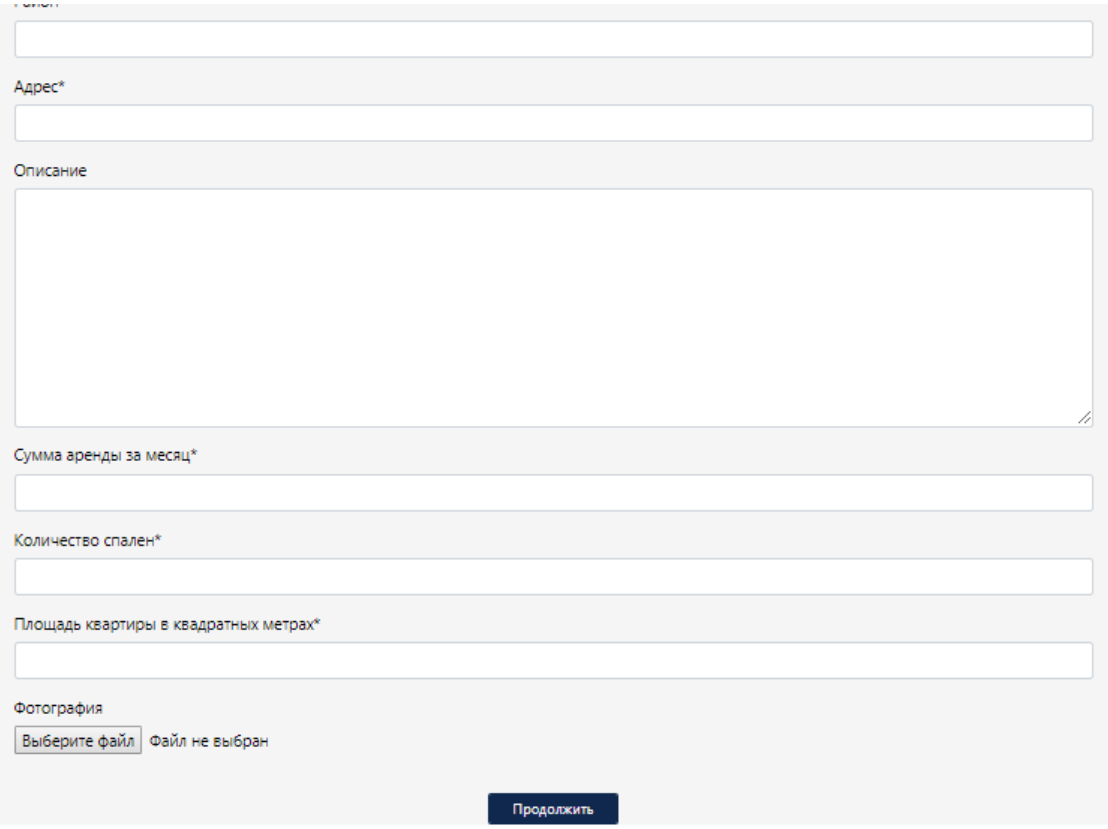

Рисунок 19 – Добавление объявления об аренде на сайт

Добавляя объявление, оно записывается в базу данных, в таблицу объявлений.

Для того, чтобы пользователи, которые хотят заключить сделку об аренде жилья, смогли увидеть это объявление потребуются некоторые действия от арендодателя. На рисунке 20 показана страница профиля арендодателя.

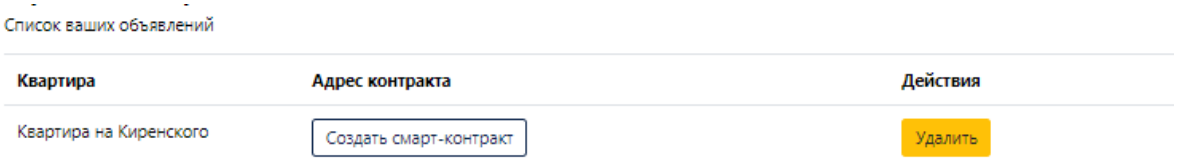

Рисунок 20 – Профиль арендодателя с созданным объявлением

3.3 Реализация второго функционального блока

Алгоритм децентрализованной системы был написан на языке Solidity, из себя он представляет смарт-контракт, который выполняет основные функции отношений по аренде жилья. Такие как: начало самой сделки, оплата за

промежуток времени, закрытие контракта, функции по проверке сведений об арендаторе и арендодателе, проверка статуса сделки. На рисунке 21 приведен рисунок показывающий структуру смарт-контракта.

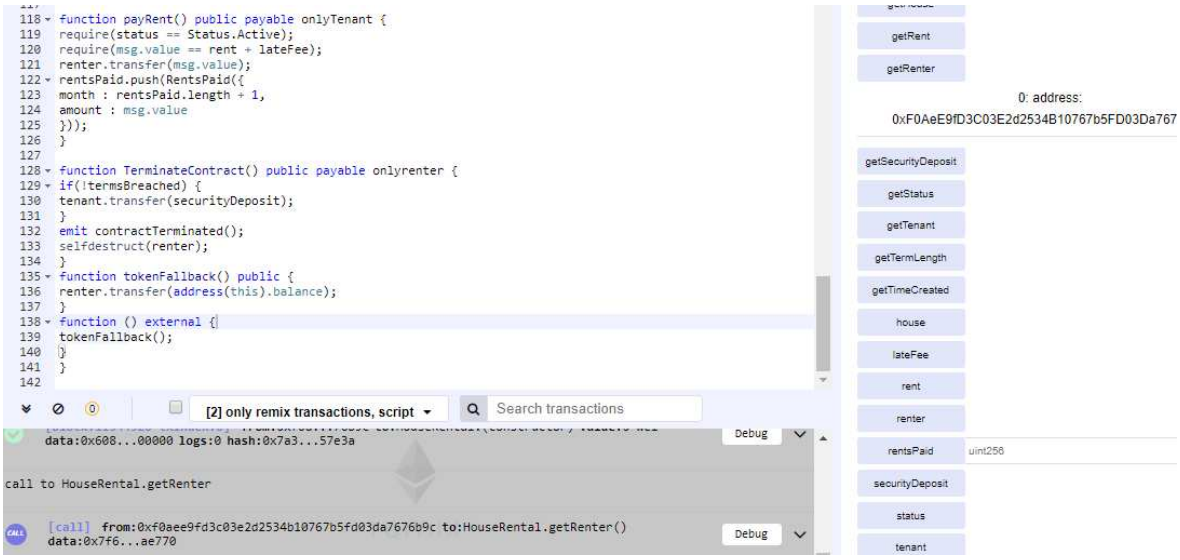

Рисунок 21 – Смарт-контракт в среде разработки Remix

Вернёмся к концу описания первого функционального блока в пункте 3.2. В профиле отображается созданная квартира и существуют две кнопки, создать смарт-контракт и удалить. Для того, чтобы пользователи смогли увидеть эту квартиру в разделе объявлений, арендодатель должен инициировать смарт-контракт, нажимая на кнопку «Создать смарт-контракт», происходят действия второго функционального блока, а, то есть взаимодействие с децентрализованной системой. На рисунке 22 предоставлен программный код, который обрабатывает действия после нажатия данной кнопки.

```
kovan_url = 'https://kovan.infura.io/v3/050656a9183d4fd7bcac660f237a1724'
web3 = Web3(Web3.HTTPProvider(kovan_url))
print(request.user.eth_acc)
web3.eth.defaultAccount = Web3.toChecksumAddress(request.user.eth_acc)

abi = json.loads('[{"constant":false,"inputs":[],"name":"TerminateContract","output
bytecode = '6080604052620f4240600c55620f4240600d553480156200001f57600080fd5b5060405

private_key = form.cleaned_data['private_key']
RentalAgreement = web3.eth.contract(abi=abi, bytecode=bytecode)

acct = web3.eth.account.privateKeyToAccount(private_key)

rentPrice = object.price
rentLength = form.cleaned_data['rentLength']
rentHouse = object.title

tx_build = RentalAgreement.constructor(rentPrice, rentHouse, rentLength).buildTrans
    {'from': acct.address,
     'nonce': web3.eth.getTransactionCount(acct.address), })

signed = acct.signTransaction(tx_build)

tx_hash = web3.eth.sendRawTransaction(signed.rawTransaction)
```

Рисунок 22 – Алгоритм соединения и вызова конструктора смарт-контракта

Алгоритм реализует основные возможности библиотеки web3.py.

Изначально алгоритмы были протестированы на программном обеспечении Ganache, имитирующем работу децентрализованной сети. На рисунке 23 показан терминал ПО Ganache, на нём отображается выполненная транзакция по созданию контракта.

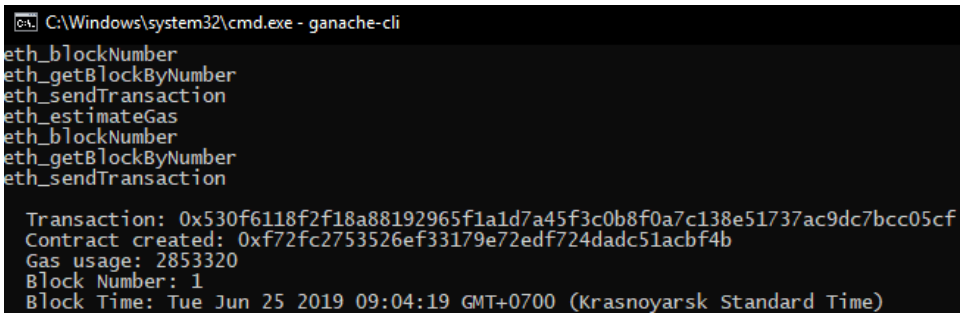

Рисунок 23 – Терминал программы Ganache

Для пользователя создание самого контракта выглядит, как окно, которое просит ввести приватный ключ от аккаунта в сети Ethereum. После этого конструктор инициализирует смарт-контракт, с помощью приватного ключа происходит его подпись и отправка в децентрализованный узел. Все адреса, полученные во время создания контракта, обрабатываются и отображаются на

сайте, на рисунке 24 показан интерфейс профиля, когда смарт-контракт был инициализирован.

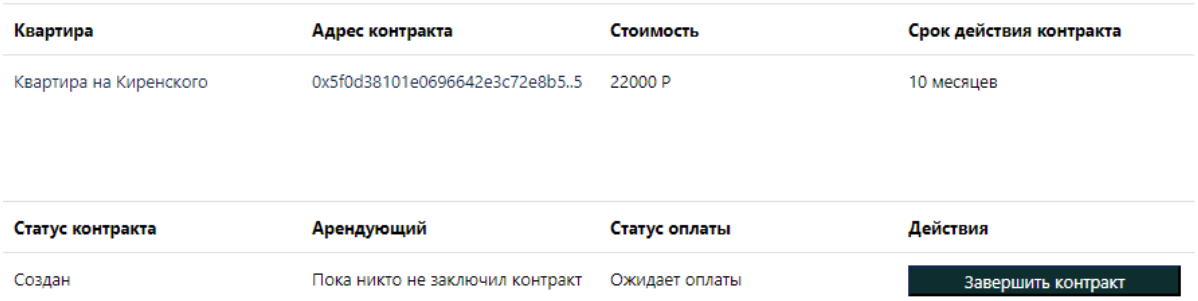

Рисунок 24 – Профиль арендодателя после создания смарт-контракта

Арендодатель сможет увидеть адрес контракта в сети, статус контракта, основную информацию по сделке, адрес арендующего в сети, а также кнопку, которая позволит завершить действие контракта.

Для того, чтобы действительно проверить работоспособность системы, каждый адрес изображенный на странице представлен ссылкой и ведёт на сайт Etherscan.io. Который отображает всем пользователям по всему миру, что контракт был создан. На рисунке 25 можно увидеть скриншот с сайта, который показывает информацию о контракте.

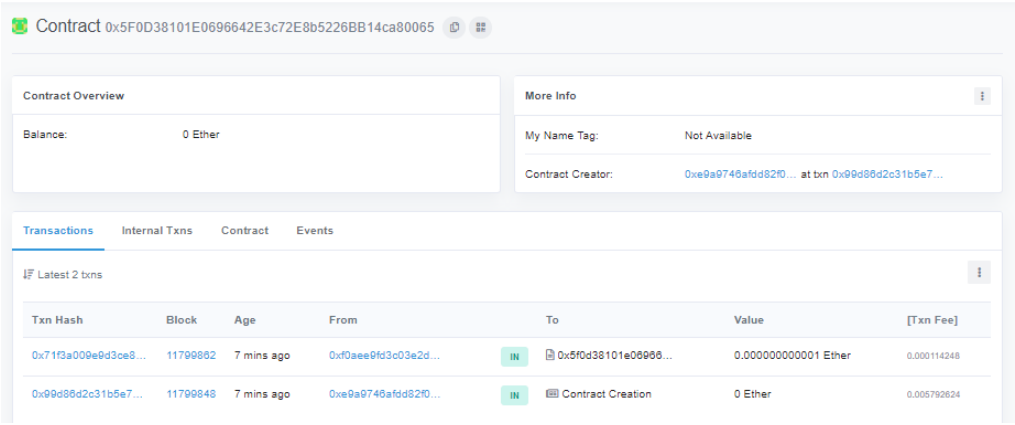

Рисунок 25 – Информация о контракте на сайте Etherscan.io

Так как смарт-контракт создан, введённая булева переменная is_published поменяет своё значение на True, что позволит созданному объявлению

отобразится на странице с последними объявлениями, оно полностью соответствует тому, как её заполнял арендодатель. Страница полного объявления содержит в себе описание, фотографию, информацию об арендодателе и кнопку снять в аренду. Эту кнопку будет использовать арендующий, чтобы подтвердить, что именно он будет лицом, участвующим в контракте. Нажав на эту кнопку, выйдет всплывающее окно, в которое пользователь должен ввести приватный ключ и сообщение для арендодателя. На рисунке 26 показана страница объявления с открытым окном заключения контракта.

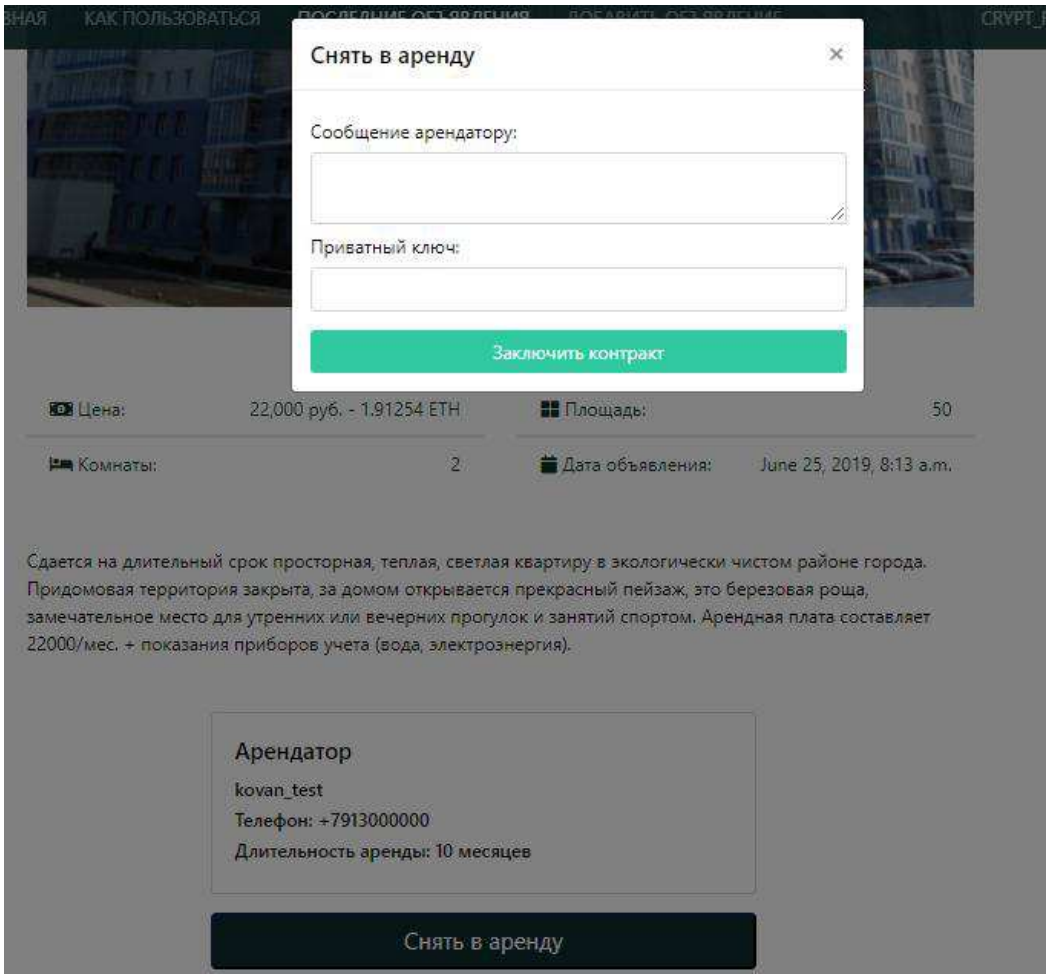

Рисунок 26 – Окно заключения контракта пользователем

Нажимая кнопку заключить контракт, происходит операция похожая на то, когда арендодатель создавал смарт-контракт. Начинается вызов и

построение функции смарт-контракта, её подпись с помощью приватного ключа пользователя и отправка в децентрализованный узел.

После этого обновляется профиль арендующего, появляется таблица с данными полученными в процессе отправления транзакции.

На рисунке 27 можно увидеть, как меняется профиль пользователя после того как он заключил контракт с арендодателем.

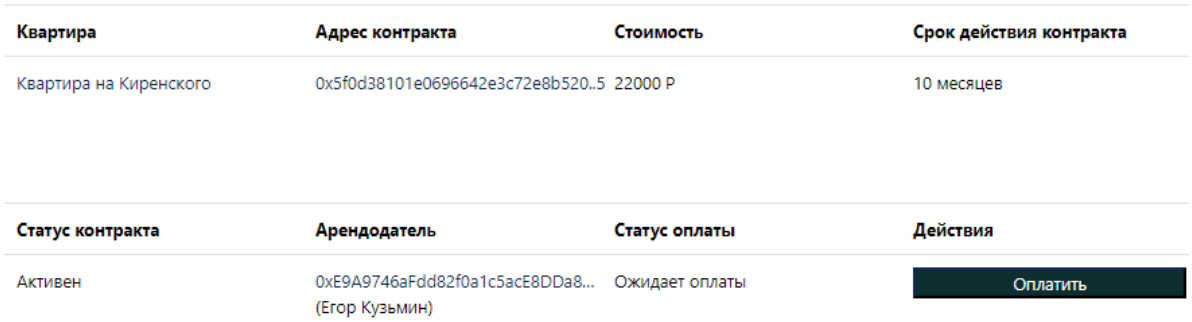

Рисунок 27 – Профиль арендующего на сайте

Профиль похож на профиль арендодателя, но здесь меняется кнопка действия. Новая кнопка позволяет пользователю оплачивать месячную аренду, посредством функций, которые вызываются из смарт-контракта. Нажимая кнопку оплатить, пользователь снова вводит приватный ключ, происходит конвертация суммы указанной арендодателем в эфир (eth – валюта сети Ethereum). Алгоритм начинает выполнение функции, на рисунке 28 можно увидеть код, в котором есть: построение транзакции – buildTransaction(), подписание транзакции – signTransaction() и отправление – sendRawTransaction().

```
tx_hash = contract.functions.payRent().buildTransaction(
    {'from': acct.address, 'value': web3.toWei(price, 'wei'),
     'nonce': web3.eth.getTransactionCount(acct.address), })

signed = acct.signTransaction(tx_hash)

tx_hash = web3.eth.sendRawTransaction(signed.rawTransaction)
```

Рисунок 28 – Функция смарт-контракта по оплате аренды

После подписи и отправки этой функции остаётся транзакционный хэш, в котором (с помощью etherscan.io) можно посмотреть между какими адресами был произведен обмен криптовалютой. В то же время в профилях меняется статус оплаты, который подсчитывает, сколько раз производилась оплата. Также под подсчётом будет указан транзакционный хэш последней транзакции, а на почту арендодателя придёт уведомление об оплате. На рисунке 29 показано, как меняется статус оплаты на сайте.

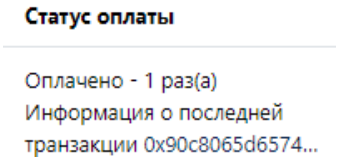

Рисунок 29 – Смена статуса оплаты и ссылка на хэш

Для дальнейшего подтверждения прозрачности операций в сети на рисунке 30 показана информация о транзакции, в которой была произведена оплата. Линией выделена строка, которая даёт информацию о том куда был произведён трансфер криптовалюты. Два адреса - это адреса арендующего и арендодателя соответственно.

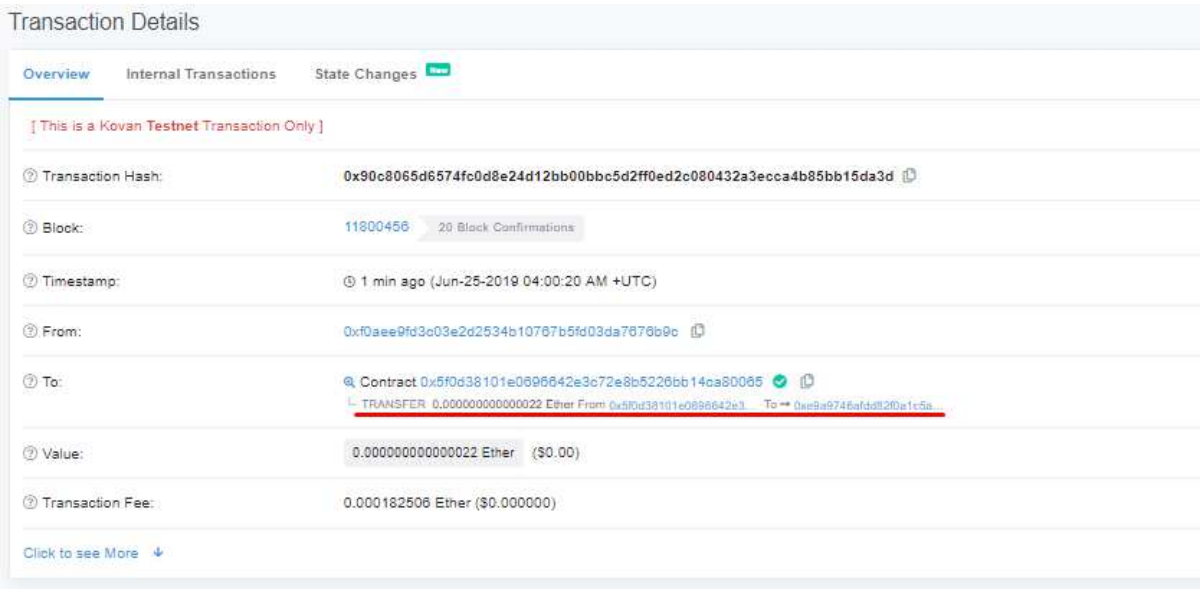

Рисунок 30 – Детали транзакции об оплате аренды на сайте Etherscan.io

Для того, чтобы закрыть контракт, в случае нарушении правил пользования квартирой, либо по другим причинам, используется кнопка завершить аренду в профиле арендодателя. Вызывается деструктор контракта. До того, как контракт перестанет действовать происходит перевод криптовалюты обратно арендующему. Вместе с этим удаляются записи из локальной базы данных о контракте. Статус объявления меняется на неопубликованное и арендодатель снова сможет выставить свою квартиру в аренду.

На странице арендующего будет отображено сообщение о статусе сделки, а также ссылка на транзакцию деструктора, для того, чтобы пользователь убедился в правильности данных. Кнопка удалить позволяет пользователю удалить эту запись. На рисунке 31 показан профиль арендующего после завершения действия контракта.

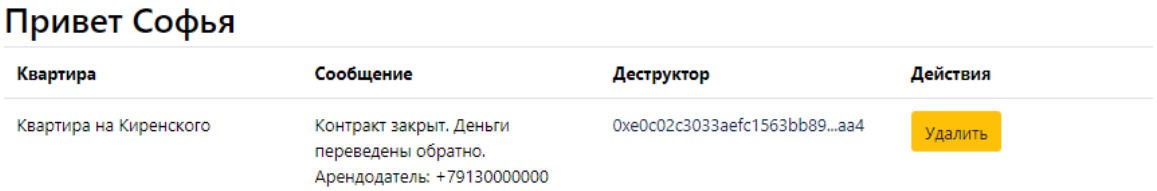

Рисунок 31 – Профиль арендующего после закрытия контракта

3.4 Анализ полученных результатов

Разработанное приложение показывает, что интеграция децентрализованной системы была проведена.

Доказать повышение безопасности не просто, всё упирается в физические носители и человеческий фактор. Любой договор об аренде, непонимание людей может нарушить условия и принести ущерб одной из сторон. Децентрализованная система не допустит такого.

По ходу выполнения работы стало понятно, что приватный ключ является почти полным заменителем согласия пользователя на произведение каких-то

операций, а открытость и прозрачность децентрализованной системы, доступность её со всех уголков мира, невозможность поменять и внести изменения, доказывают, что такое использование целесообразно и более безопасно.

ЗАКЛЮЧЕНИЕ

Итогом бакалаврской работы стала интеграция децентрализованной системы в процесс аренды жилья. Функции обычного веб-приложения и использование новых технологий, основанных на криптовалютах и криптографии, проложили возможность для осуществления данной задачи.

Помимо этого, было решено множество задач, без которых цель работы была бы не осуществима. Проведен анализ предметной области, выявлены основы построения криптовалют. Был создан план по разработке системы состоящей из двух блоков. Разработано приложений показывающее работоспособность интеграции децентрализованной системы, проведено быстрое тестирование системы.

Результаты показали, что использование децентрализации делает систему открытой и прозрачной, а, следовательно, безопаснее.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Блокчейн для чайников: книга // Лоуренс Тиана. – Москва: Изд–во «ДИАЛЕКТИКА», 2018. – 272 с.
2. How does the Blockchain Work? [Электронный ресурс] // Collin Thompson on Medium, 2016. – Режим доступа: http://medium.com
3. Осваиваем биткойн. Программирование блокчейна: книга // Антонопулос Андреас М.. – Москва: Изд-во «ДМК Пресс» , 2018 –695 с.
4. CASE-технологии - Практикум: книга // Федотова Д.Э., Семенов Ю.Д., Чижик К.Н. – Москва: Изд-во «Горячая Линия - Телеком» , 2005 –165 с.
5. Mastering Ethereum: Building Smart Contracts and DApps: книга // Andreas Antonopoulos and Gavin Wood Ph.D. – London: 2018 –424 с.
6. СТАНДАРТ ОРГАНИЗАЦИИ Система менеджмента качества. Общие требования к построению, изложению и оформлению, документов учебной деятельности СТО 4.2-07-2014 – Красноярск: ИПК СФУ, 2014. – 60 с
7. Mastering Bitcoin: Programming the Open Blockchain 2nd Edition: книга // Андреас Антонопулос, Калифорния, США: O'Reilly Media, 2017. –40-85с.
8. Mastering Ethereum: Building Smart Contracts and Dapps: книга // Андреас Антонопулос, Калифорния, США: O'Reilly Media, 2017. –30-90с.
9. Криптовалюта как новый инструмент денежного рынка: доклад, тезисы доклада // О. В. Старова, М. С. Фирскина. – Красноярск Сибирский федеральный университет. – Красноярск, 2017. 148 с.
10. Трубникова Е. И. Феномен криптовалюты: характеристики, предпосылки, институциональный анализ рынка // Е. И. Трубникова // Инфокоммуникационные технологии, 2014. 74-85 с.
11. Самая твердая валюта – BitCoin. – [Электронный ресурс] // М.,2006-2018. – Режим доступа: http://moneynews.ru/Article/17893
12. Документация библиотеки Web3py [Электронный ресурс] // Piper Merriam, Jason Carver. – Режим доступа: https://web3py.readthedocs.io/en/stable/

13. Технология Blockchain. Практическое применение [Электронный ресурс]. – Режим доступа: https://blog.dti.team/tekhnologiya-blockchain-prakticheskoeprimenenie/

14. Смарт-контракты: функции и применение // Осмоловская А.С. // Иркутский государственный университет, 2018, УДК 338.681.3

15. Смарт-контракты и платформы для их реализации [Электронный ресурс]. – Режим доступа: https://ecrypto.ru/blokchejn/smart-kontrakty-i-platformy-dlya-ihrealizatsii.html

16. Документация Django [Электронный ресурс]. – Режим доступа: https://docs.djangoproject.com/en/2.2/

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
институт
Информационных систем
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ИС
_____ П.П.Дьячук
подпись инициалы, фамилия
«21» 06 2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 – «Информационные системы и технологии»

Интеграция децентрализованной системы в процесс аренды жилья

Руководитель	_____ 21.06.19	ст. преподаватель	Ю.В.Шмагрис
	подпись, дата	должность, ученая степень	
Консультант	_____ 21.06.19	к.т.н., доцент	И.А.Легалов
	подпись, дата	должность, ученая степень	
Выпускник	_____ 21.06.19		Е.К.Кузьмин
	подпись, дата		
Нормоконтролер	_____ 21.06.19	ст. преподаватель	Ю.В.Шмагрис
	подпись, дата	должность, ученая степень	

Красноярск 2019