ФИРСОВА С. А., ВДОВИН П. С., КАРПУШКИНА И. С.

ПРИМЕНЕНИЕ ТЕХНОЛОГИИ БЛОКЧЕЙН ДЛЯ ДИСТАНЦИОННОГО ГОЛОСОВАНИЯ

Аннотация. В статье рассматривается созданный авторами прототип программной системы, предназначенной для принятия коллегиальных решений путем проведения дистанционного голосования на основе технологии блокчейн. Реализовано как открытое, так и тайное голосование, автоматизировано получение отчетной документации по результатам голосования, а также предусмотрена возможность генерации протоколов заседаний, выписок из протоколов и т. п.

Ключевые слова: прототип программной системы, системы дистанционного голосования, технология блокчейн, диаграммы UML.

FIRSOVA S. A., VDOVIN P. S., KARPUSHKINA I. S.

APPLICATION OF BLOCKCHAIN TECHNOLOGY FOR REMOTE VOTING

Abstract. The article presents a software system prototype developed by the authors and designed for making collective decisions by remote voting based on blockchain technology. As a result, both open and secret voting have been implemented. The receipt of reporting documents based on voting results has been automated. It is also possible to generate minutes of meetings, extracts from minutes, etc.

Keywords: software system prototype, remote voting systems, blockchain technology, UML diagrams.

Введение. Дистанционное голосование – процесс принятия решения, путём обычного голосования, с применением специальных электронных средств голосования и технических электронных средств для подсчета голосов и оглашения результата.

Проанализировав существующие программные решения для систем дистанционного голосования, можно сформулировать следующие недостатки:

- недостаточная прозрачность процесса голосования;
- недостаточная отказоустойчивость системы;
- недостаточная устойчивость к взлому.

Поэтому всё чаще для программных систем подобного назначения предлагается использовать протокол на основе технологии блокчейн, который поможет устранить сформулированные выше недостатки.

Блокчейн – это распределенная база данных, которая может храниться на рабочих станций, географически удаленных друг от друга. Блокчейн можно рассматривать как цепочку блоков с тремя столбцами, где каждая строка представляет отдельную транзакцию (операцию), которую совершил участник сети. В первом столбце хранится метка времени о транзакции; во втором – сведения о транзакции; третий столбец содержит в себе свой хэш и хэш предыдущего блока, так все блоки представляют собой единую цепочку, то есть историю всех действий в системе с начала её старта. Транзакция является основным событием в технологии блокчейна, которое разрешено базовым протоколом. При проведении этого события идет обновление информации в блоке цепочки, передающееся к блокчейн-сети по всему миру.

Среди базовых принципов технологии блокчейн выделяют:

– распределенность – нет единой централизованной базы данных, взломав которую, можно удалить или фальсифицировать информацию;

– безопасность – даже если удастся взломать несколько блоков и изменить в них данные, то безопасность системы не пострадает. Для изменения блокчейн-системы необходимо взломать все блоки, что маловероятно. Любая попытка взлома будет замечена участниками сети, кроме того, система защищена шифрованием;

– прозрачность – вся база данных открыта, посмотреть данные блока может любой желающий, т.е. увидеть транзакцию может каждый, однако узнать непосредственных участников транзакции будет возможно, только если они сами пожелают обнародовать этот факт;

– монолитность – технология дает возможность обмениваться данными напрямую между отправителем и получателем. Подлинность операций в системе проверяется исключительно участниками транзакции.

Архитектура прототипа программной системы для проведения дистанционного голосования. Авторами статьи был создан прототип программной системы, предназначенной для принятия коллегиальных решений путем проведения дистанционного голосования на основе технологии блокчейн. Данный прототип позволяет проводить дистанционное голосование (открытое и тайное), а также генерировать различную отчетную документацию по результатам голосования для заседаний кафедр, советов факультетов, ректората и т.п. в Мордовском государственном университете.

Архитектура разработанного прототипа описана с помощью диаграмм UML, при этом можно выделить следующие компоненты спецификации системы: концептуальную модель, модель размещения, модель реализации.

Для описания концептуальной модели была разработана диаграмма вариантов использования:

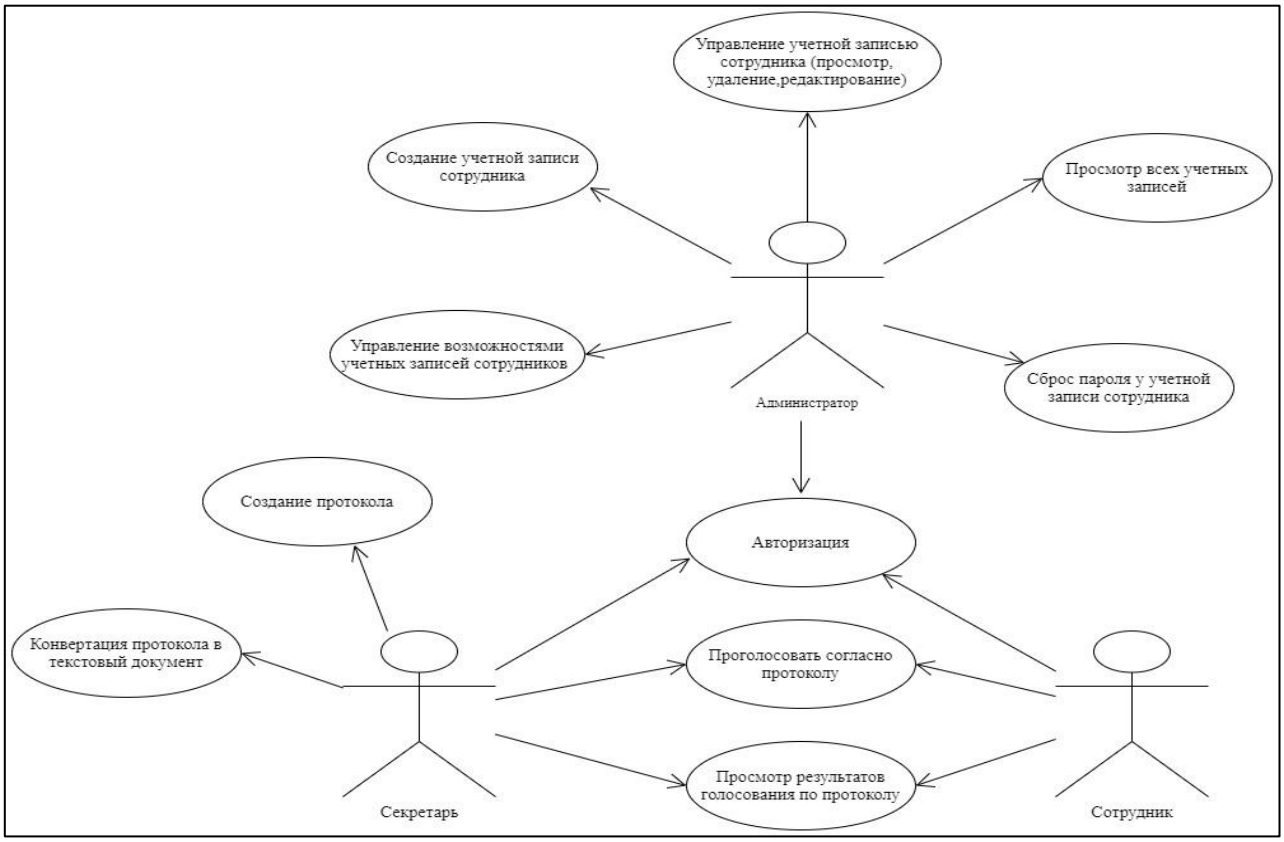


Рис. 1. Диаграмма вариантов использования.

В результате моделирования были выявлены следующие актеры:

– администратор – пользователь, осуществляющий управление учетными записями сотрудников;

– секретарь – пользователь, выполняющий функции создания заседаний пользователей, осуществляющий работу с протоколами голосования, а также формирующий различные выписки из протоколов заседаний;

– сотрудник – пользователь, осуществляющий голосование согласно протоколу.

Модель размещения описывает варианты физического размещения элементов системы, для представления этой модели удобно использовать UML-диаграмму развертывания (см. рис. 2).

Как видно из диаграммы развёртывания – результаты голосования по протоколам передаются на сервер, а затем сохраняются в базе данных. База данных создана и в настоящий момент управляется СУБД MS SQL Server 2019.

При тайном голосовании результаты помещаются в хранилище блокчейн- платформы Waves, а затем считываются программой сервера и дублируются в базу данных.

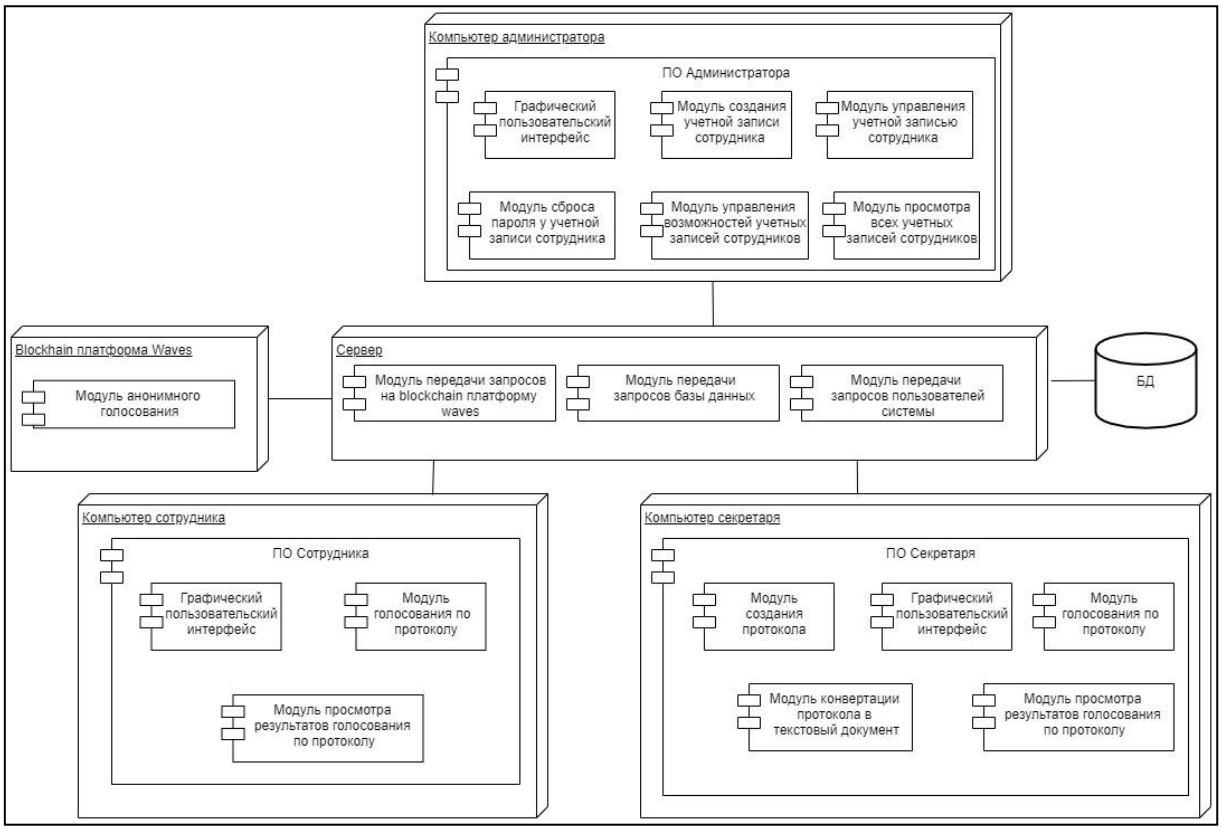


Рис. 2. Диаграмма развертывания.

Модель реализации описывает разделение системы на отдельные компоненты, независимые задачи, подпрограммы, информационные и управляющие потоки и связи между элементами системы. Для представления модели реализации используется диаграмма компонентов, представленная на рисунке 3.

Архитектура программной системы для принятия коллегиальных решения делится на три уровня:

– уровень представления информации;
– уровень логики;
– уровень данных.

На уровне представления информации осуществляется взаимодействие системы с пользователем. Через графический пользовательский интерфейс производится ввод данных в систему и вывод запрашиваемой пользователем информации.

Все вычисления системы производятся на уровне логики. Здесь обрабатываются команды и запросы, принимаемые от пользователя системы.

Уровень данных содержит базу данных, в которой находятся все данные о пользователях системы, протоколах, результатов голосований по протоколам. Также в этот уровень входит хранилище данных блокчейн-платформы Waves, которое содержит результаты при тайном голосовании.

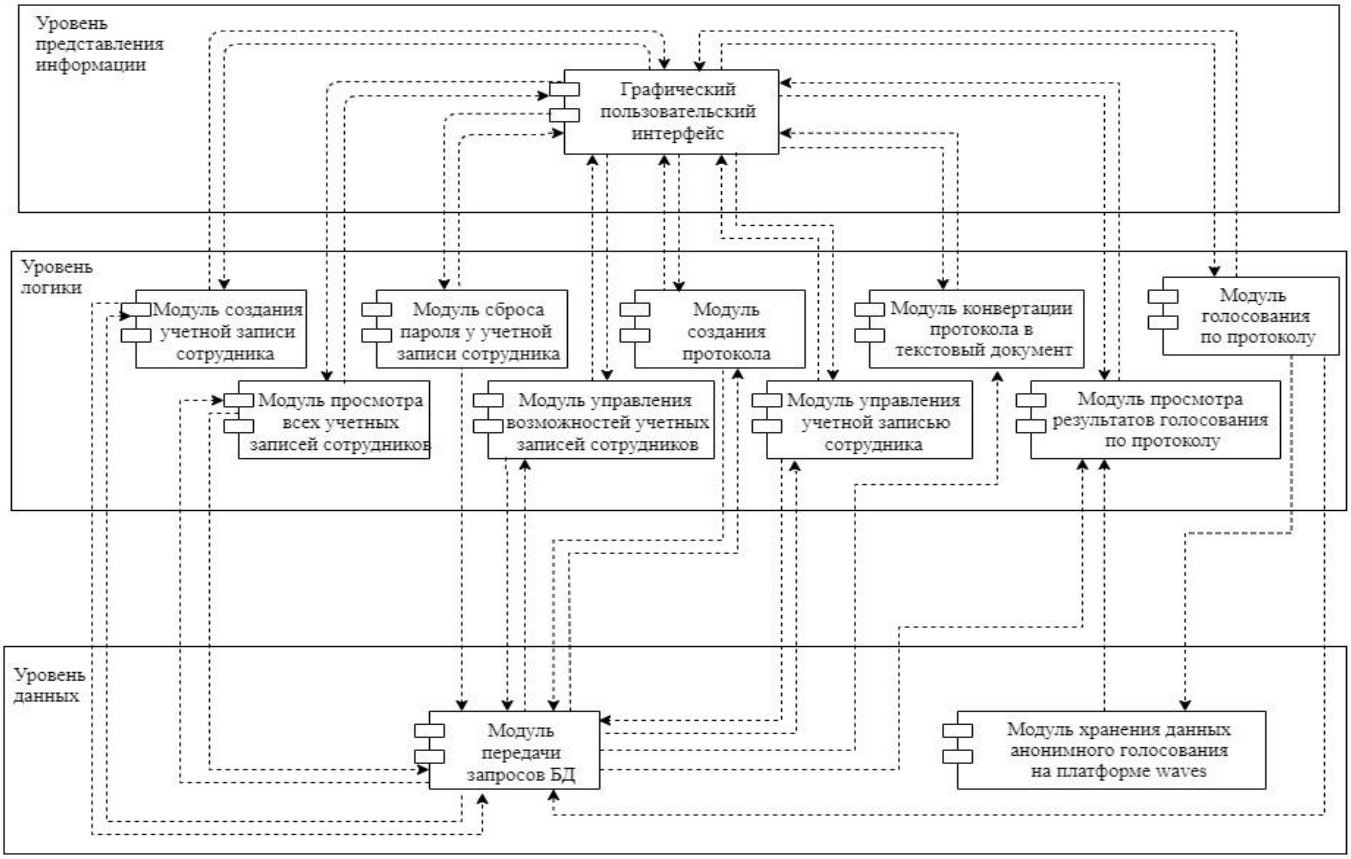


Рис. 3. Диаграмма компонентов.

Для хранения информации системы была выбрана система управления базами данных (далее СУБД) MS SQL Server 2019. База данных прототипа программной системы для принятия коллегиальных решений содержит 5 таблиц:

– «users» (Пользователи): содержит сведения о пользователях системы;

– «info_protocol» (Протоколы): содержит подробную информацию о протоколах;

– «answers_protocol» (Ответы на протоколы): содержит ответы пользователей на вопросы, по которым проходит голосование по протоколам;

– «user_protocol» (Доступ для пользователей): содержит информацию о протоколах, которые доступны определённому пользователю;

– «result_protocol» (Результаты голосования): содержит результаты по завершившимся голосованиям.

Схема базы данных представлена на рисунке 4.

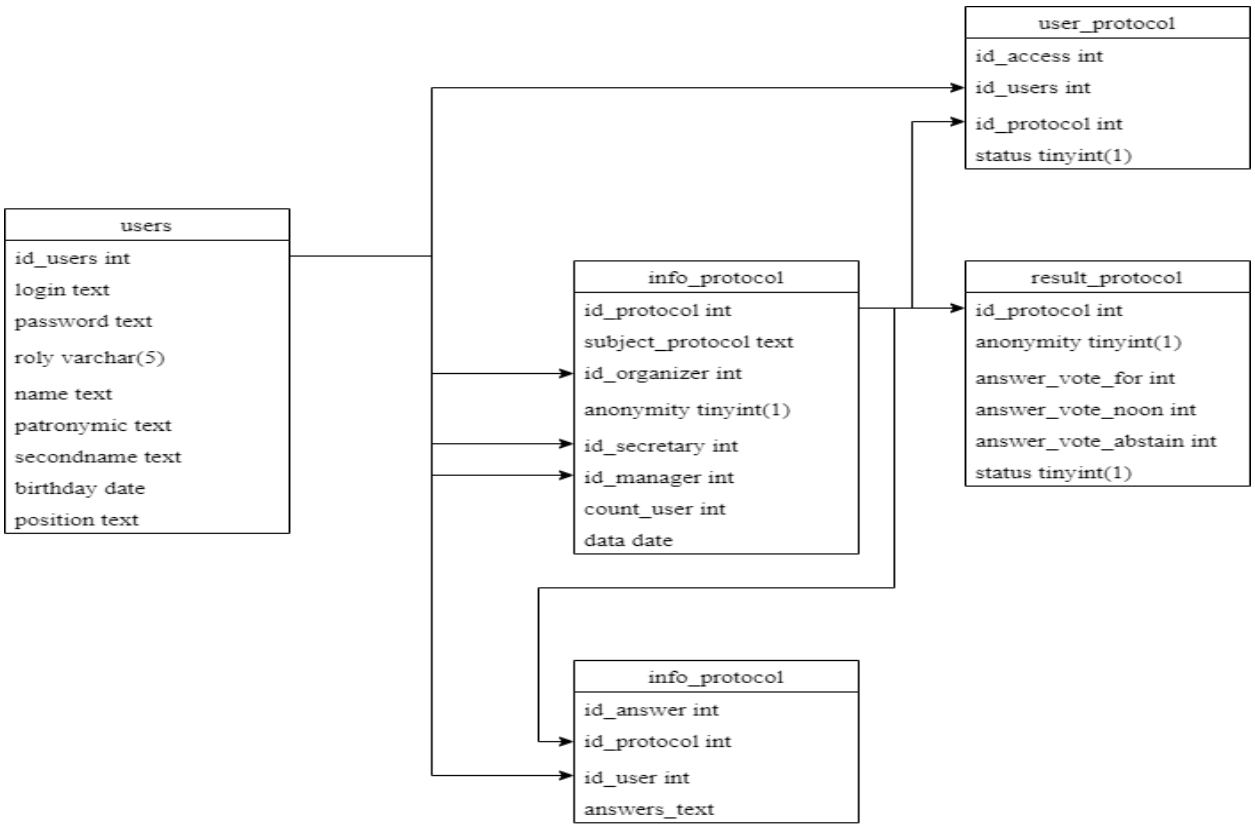


Рис. 4. Схема базы данных.

Технологии, использованные при разработке прототипа. При разработке прототипа программной системы использовались технология Windows Forms, платформы Microsoft.NET Framework и технология блокчейн на платформе Waves.

В настоящее время существует значительное количество реализаций блокчейнов, среди которых присутствуют такие платформы, как Bitcoin, Waves, Ethereum, zCash, Ripple, Stellar, IOTA и др. Каждая из этих платформ имеет свои положительные черты, но при разработке децентрализованных приложений следует обратить своё внимание на платформу Waves, поскольку платформа имеет тестовый режим блокчейна, и свою собственную среду разработки.

При разработке децентрализованного приложения можно выделить 2 основных этапа: написание смарт-контракта, развертывание смарт-контракта в блокчейн сети Waves. Смарт-контракт играет ключевую роль в разработке модуля, на нем описываются все функции, которые будут позволять пользователю выполнять те или иные манипуляции с данными.

Смарт-контракт – протокол взаимодействия между пользователями, пользователем и компьютером или компьютером и компьютером в соответствии с правилами, определенными в исходном коде смарт-контракта.

Waves IDE – онлайн-среда для разработки и тестирования смарт-контрактов на языке Ride. Основными возможностями Waves IDE являются:

– наличие библиотеки смарт-контрактов и тестов;

– компиляция и установка Ride-скриптов;

– подписание и отправка транзакций;

– наличие интерактивной консоли JavaScript со встроенными функциями работы с блокчейном;

– запуск JavaScript-тестов;

– отправка ссылки на файл.

Работа с Waves IDE. Для разработчиков платформа Waves создала тестовую блокчейн-сеть, для откладки своих смарт-контрактов. Основной особенностью тестовой сети является то, что не нужно разворачивать узел блокчейн-сети на своей виртуальной машине. Тестовая блокчейн-сеть в связке с IDE Waves является отличным инструментом разработки смарт-контракта.

Для начала разработки необходимо создать новый аккаунт в тестовой блокчейн-сети Waves, после чего создается смарт-контракт, в котором прописываются основные функции.

Основная функция будущего смарт-контракта – это функция верификации транзакций, которая позволяет обезопасить смарт-контракт от несанкционированного доступа. Для этого используется аннотация Verifier языка Ride. В аргументе аннотации находится переменная, которая содержит информацию о транзакции. После аннотации реализуется функция авторизации ferify, обычно в ней разрешают определенные типы транзакций, и происходит проверка публичного ключа, если транзакция типа SetScriptTransaction, пытается внести изменений в код смарт-контракта, которой расположено в блокчейн-сети.

```
@Verifier(tx)
 func ferify() =
 {  match (tx)
    {case t: SetScriptTransaction => sigVerify(tx.bodyBytes,tx
.proofs[0], ownerpublickey)
     case c: DataTransaction => true
     case d: InvokeScriptTransaction => true
     case   => false}}
```

Рис. 5. Функция верификации транзакций ferify.

После успешной верификации транзакции, нужно её обработать, для этого используем аннотацию Callable. В аргументе аннотации, находиться переменная, которая содержит информацию о верифицированной транзакции. После аннотации, реализуется функция обработки транзакции vote, она содержит единственный аргумент, типа string, в которой будет размещен номер протокола, и выбора ответа на него. Функция определяет,

голосовал ли раньше пользователь, если да, то функция выводит сообщение об ошибке, и не записывает ответ. Если пользователь не голосовал ранее, то функция записывает его ответ в Data Transactions. Смарт-контракт оплачивает комиссию пользователя, при отправке транзакции.

```
Callable(InvokeScriptTransaction)
func vote(str: String) =
{let dataFromStorage = this.getString
(InvokeScriptTransaction.callerPublicKey.toBase58String())
if (dataFromStorage.isDefined())
    then throw("Вы уже голосовали в данном опросе!")
    else
{ScriptResult(WriteSet([DataEntry
(InvokeScriptTransaction.callerPublicKey.toBase58String(), str)]),
TransferSet([ScriptTransfer
(InvokeScriptTransaction.caller, InvokeScriptTransaction.fee, unit
)]))}}
```

Рис. 6. Функция голосования vote.

Для развертывания смарт-контракта в тестовую блокчейн-сеть Waves, нам понадобится некоторое количество токенов Waves, чтобы оплатить комиссию при отправке транзакции. Разработчики платформы решили эту проблему, создав инструмент Waves Explorer, где можно бесплатно получить токены, и просмотреть все транзакции связанные с смарт-контрактом.

Для получения токенов нужно войти в меню инструмента, выбрать тестовую сеть «Testnet», в списке инструментов выбрать «Faucet», как показано на рисунке 7.

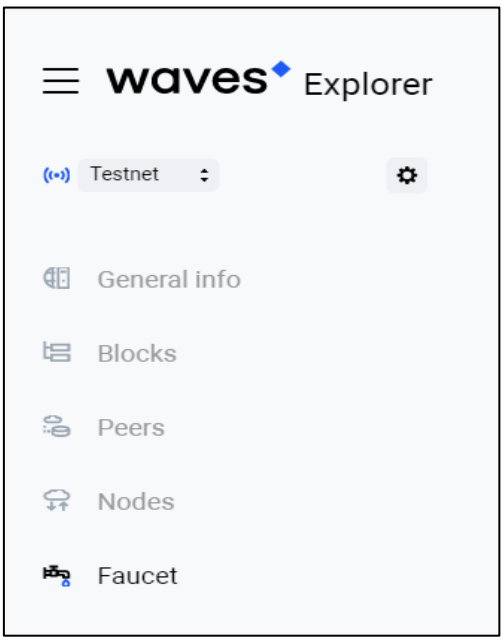


Рис. 7. Получение токенов в инструменте Waves Explorer.

После проделанных манипуляций открывается окно, в котором нужно ввести адрес смарт-контракта (см. рис. 8). Адресом является адрес аккаунта в тестовой блокчейн-сети Waves. После ввода адреса и проверки пользователя на робота, нужно нажать на кнопку «Request 10 Waves».

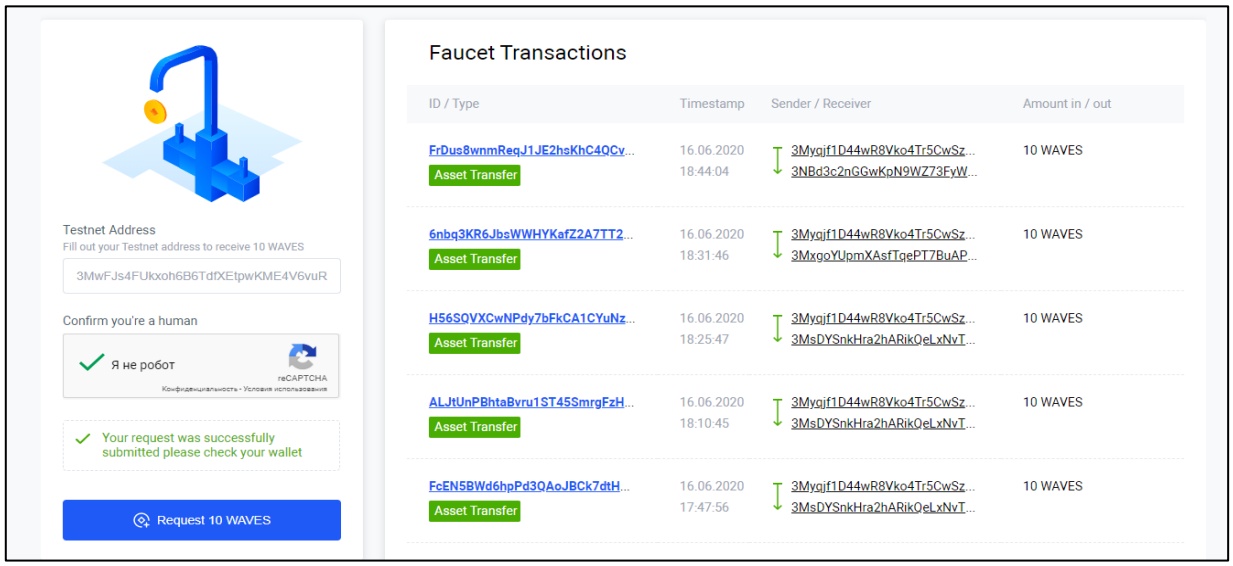


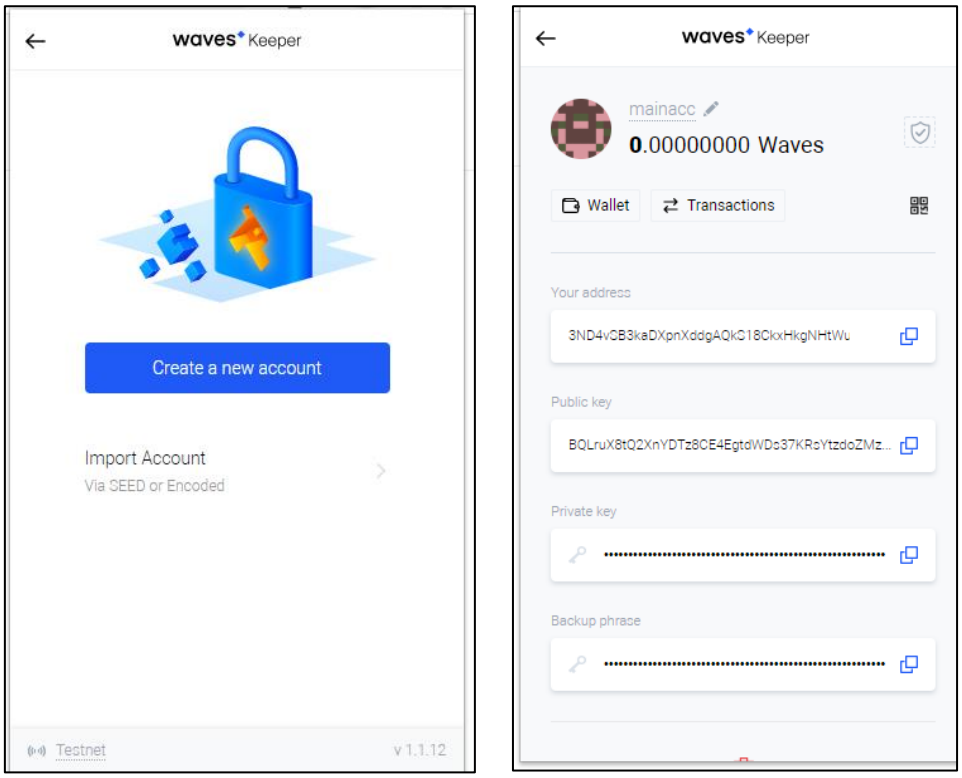
Рис. 8. Успешное получение токенов в инструменте Waves Explorer.

Полученные токены отразятся на аккаунте Waves в IDE.

После получения токенов, возможно развернуть созданный смарт-контракт в тестовую блокчейн-сеть Waves. Для этого нужно нажать на кнопку «Deploy» в IDE Waves, после нажатия открывается окно с подробным описанием транзакции, которая содержит всю основную информацию о смарт-контракте.

Для отправки транзакции следует с начало подписать транзакцию, для этого нужно нажать на кнопку «Add Sign», после этого для публикации транзакции в блокчейн-сеть необходимо нажать на кнопку «Publish». Далее при успешном развертывании транзакции в блокчейн-сеть появляется соответствующее уведомление.

Для проверки функциональности создадим тестовый протокол с тайным голосованием на несколько сотрудников. Для сотрудников нужно создать личные аккаунты на блокчейн-платформе Waves. Для этого существует браузерное расширение Waves Keeper, создание аккаунта в нем показано на рисунке 9а. Для дальнейшего голосования понадобится Private key созданного аккаунта, он находится в блоке с основной информацией об аккаунте (см. рис. 9б).



а) б)

Рис. 9. а) создание аккаунта в Waves Keeper; б) основная информация созданного аккаунта.

Проголосуем с личных аккаунтов этих сотрудников. После окончания голосования, перейдём в инструмент Waves Explorer, и просмотрим всю информацию (см. рис. 10), которая находится в смарт-контракте, для этого нажмём на кнопку «Date».

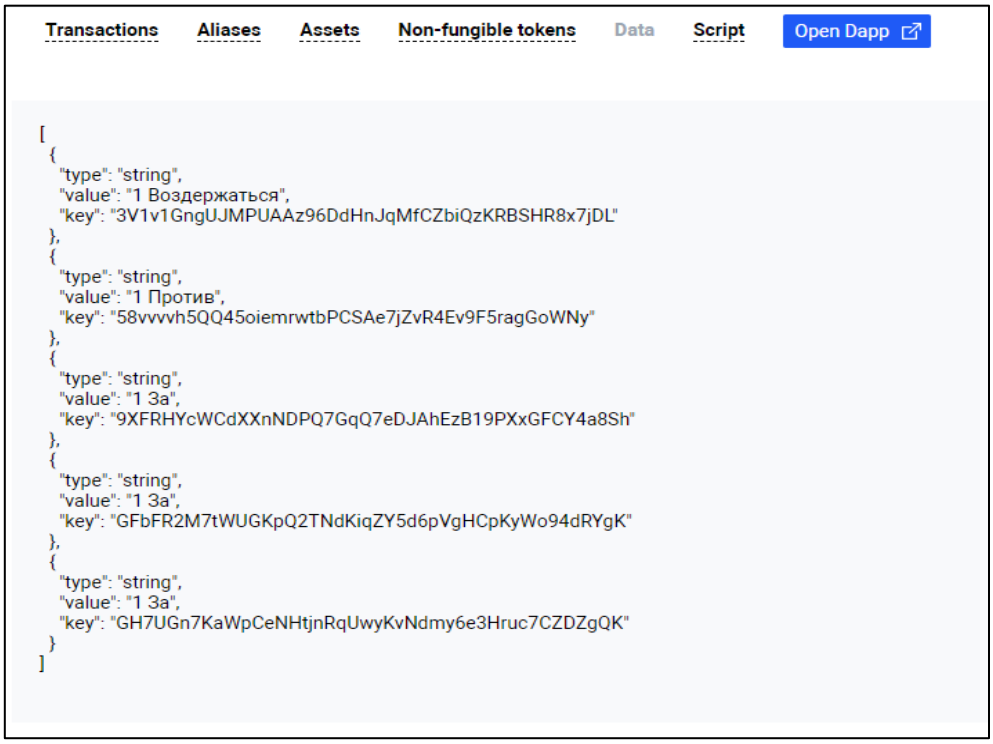


Рис. 10. Результат тестового голосования.

Заключение. Разработанный прототип программной системы позволяет проводить подразделениям университета открытое и тайное голосование по различным вопросам в дистанционном режиме и автоматизировать получение отчетной документации по результатам голосования, а также генерировать протоколы заседаний, выписки из протоколов и т.п.

СПИСОК ЛИТЕРАТУРЫ

1. Генкин А. Блокчейн и электронное голосование: о технологиях децентрализованной демократии // Самоуправление. – 2017. – № 2 (107). – С. 44–48.

2. Кутейников Д. Л. Особенности применения технологий распределенных реестров и цепочек блоков (блокчейн) в народных голосованиях // Актуальные проблемы российского права. – 2019. – № 9 (106). – С. 41–52.

3. Алексеев Р. А., Абрамов А. В. Проблемы и перспективы применения электронного голосования и технологии избирательного блокчейна в России и за рубежом // Гражданин. Выборы. Власть. – 2020. – № 1 (15). – С. 9–21.