# ВВЕДЕНИЕ

В последнее время информационные технологии шагнули так далеко вперед, что с каждым годом обстановка в мире начинает соответствовать критериям информационного общества. Все большая часть населения задействована в производстве информационных услуг, а производство товаров становится автоматизированным. При таких условиях государство вынуждено подстраиваться под развивающегося гражданина, создавать сервисы и системы, при которых поиск информации, оплата каких-либо начислений и подтверждение данных при поступлении на работу осуществлялись бы дистанционно.

В пример можно привести портал Госуслуг, который позволяет пользователю не только просматривать персональные данные, но и дистанционно оплачивать штрафы и налоги, записываться на прием к врачу, оформлять различные виды документов.

Одновременно с этим перед гражданами возникает проблема доверия тем данным, которые предоставляют им другие граждане. У работодателей все чаще возникает вопрос о подлинности документов об образовании (дипломов и сертификатов) соискателей на вакантные должности. При нынешнем уровне информатизации государственных структур работодатель вынужден запрашивать данные в Федеральной службе по надзору в сфере образования и науки или напрямую обращаться с письменным запросом в ВУЗы или другие учебные заведения. Естественно, процесс получения этих данных растягивается не на дни, а на недели и месяцы. Особенно актуальна данная ситуация для медицинских учреждений, правоохранительных органов, учебных заведений и других госучреждений.

Одной из наиболее надежных и безопасных технологий хранения данных в больших количествах в настоящее время является технология Blockchain. Наиболее перспективные направления его использования лежат в не-финансовой сфере. Многие страны уже сейчас задействуют эту технологию в системах, предназначенных для здравоохранения, документооборота, хранения и подтверждения всевозможных прав. Например, в Нидерландах, Эстонии, США внедряются единые национальные реестры медицинских карт, построенные на blockchain’е [1].

Ввиду этих мировых тенденций и сформулированной выше проблемы было принято решение о создании новой прозрачной системы, которая позволит:

- безопасно хранить данные о документах об образовании;

- быстро получать эти данные;

- гарантировать подлинность и неизменность объектов в системе.

Были поставлены следующие задачи:

- обзор аналогов и конкурентов

- проектирование и разработка децентрализованного хранилища данных

- разработка соглашений о добавлении и считывании данных из хранилища

- реализация удобного пользовательского интерфейса для доступа к данным о документах

- анализ системы с точки зрения информационного менеджмента

# АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## 1 Глоссарий

Blockchain – распределенное хранилище данных, построенное по определенным правилам объединения блоков в цепочки.

Смарт-контракт – компьютерный алгоритм, предназначенный для заключения и поддержания коммерческих контрактов в технологии блокчейн.

Ethereum – платформа для создания децентрализованных онлайн-сервисов на базе блокчейна.

Solidity – язык виртуальной машины Ethereum, предназначенный для написания умных контрактов.

ЭЦП – электронно-цифровая подпись – реквизит электронного документа, позволяющий отследить подлинность подписанного ею файла.

Документ об образовании – документ, подтверждающий прохождение соискателем определенных курсов, дипломы, сертификаты и т.д. Характеризуется следующими параметрами:

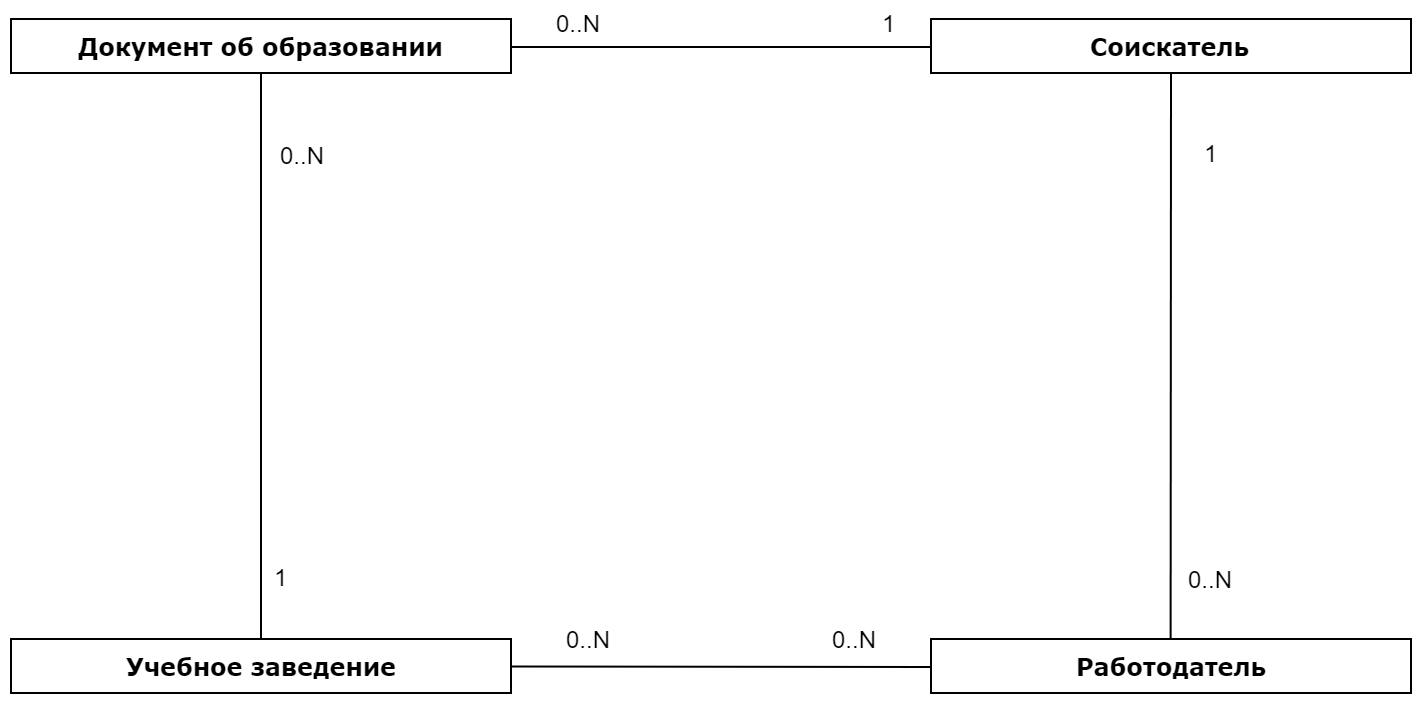
* серия и номер
* владелец
* отправитель (учебное заведение, выпустившее данный документ)
* дата выдачи
* квалификация
* специальность

Соискатель — обладатель документа об образовании, который хочет подтвердить подлинность своего диплома или т.п для получения работы. Может быть владельцем документов об образовании.

Работодатель – организация или физическое лицо, которое хочет удостовериться в подлинности документа соискателя.

Учебное заведение – заведение, обладающее правами выпускать документы об образовании. Может выступать в качестве отправителя документа.

На рис. 1 приведено взаимоотношение основных объектов предметной области.

Рисунок 1 – Концептуальная модель данных ИС ДРДО

## 2 Обзор аналогов

На основании частей 9 и 10 статьи 98, пункта 2 части 15 статьи 107 Федерального закона от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации», и постановления Правительства Российской Федерации от 26 августа 2013 г. № 729. «О федеральной информационной системе «Федеральный реестр сведений о документах об образовании и (или) о квалификации, документах об обучении», Федеральная служба по надзору в сфере образования и науки осуществляет формирование и ведение Федерального реестра сведений о документах об образовании и (или) о квалификации, документах об обучении.

Целями создания Федерального реестра являются:

* Ликвидация оборота поддельных документов государственного образца об образовании
* Обеспечение ведомств и работодателей достоверной информацией о квалификации претендентов на трудоустройство
* Сокращение числа нарушений и коррупции в образовательных учреждениях
* Повышение качества образования за счет обеспечения общественности достоверной информацией о выпускниках

В рамках проектной деятельности Рособрнадзором создана Автоматизированная система формирования и ведения ФРДО (ФИС ФРДО), обеспечивающая сбор сведений о выданных документах с образовательных учреждений, накопление этих сведений в единой базе данных. [2]

Система доступна по следующему адресу: <http://frdocheck.obrnadzor.gov.ru/>

Данная система отличается тем, что просто загрузить документ в реестр невозможно: необходимо пройти не только процедуру регистрации учебного заведения, но и получения ЭЦП. В свою очередь, ИС ДРДО позволяет упростить процесс регистрации за счет модернизированной системы прав и доверия.

Второй недостаток – полностью закрытый процесс добавления и поиска документов в федеральном реестре. Это способствует повышению уровня коррупции в данной области и увеличения числа нарушений при добавлении документов. В ИС ДРДО все транзакции записываются и доступны всем участникам блокчейна. В такой прозрачной системе легче предотвратить намеренные и случайные ошибки при добавлении документа.

И третье – следуя из того, что все транзакции (они же изменения) фиксируются в блокчейне, невозможно незаметно подменить объект. Это гарантирует подлинность внесенных данных о документе об образовании.

# ПРОЕКТИРОВАНИЕ

## 1 Составление технического задания на проектирование

2.1 Общие сведения

2.1.1. Полное наименование системы и ее условное обозначение

Информационная система «децентрализованный реестр документов об образовании»

2.1.2. Наименование предприятий (объединений) разработчика и заказчика (пользователя) системы и их реквизиты:

Инициативный проект

Компания разработчика: ИП Штых А.Д.

2.1.3. Перечень документов, на основании которых создается система, кем и когда утверждены эти документы

Разработка Технического задания проводилась с использованием следующих стандартов:

ГОСТ 34.601-90 Автоматизированные системы. Стадии создания;

ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы;

ГОСТ 34.201-89 Виды, комплектность и обозначение документов при создании автоматизированных систем (частичное использование);

ГОСТ 24.104-85 Автоматизированные системы управления. Общие требования;

ГОСТ Р 56214-2014/ISO/TS 8000-1:2011 Качество данных. Часть 1. Обзор.

2.1.4. Плановые сроки начала и окончания работы по созданию системы

Начало: апрель 2017

Конец: апрель 2018

2.2 Назначение и цели создания системы

Система предназначена для использования учебными заведениями в целях добавления данных документов об образовании в реестр, соискателями на вакантные должности и работодателями для подтверждения подлинности документов об образовании.

Система так же может использоваться другими органами, юридическими и физическими лицами для этих же целей.

Цель проектирования состоит в уменьшении времени, затрачиваемого на получение подтверждения подлинности документов об образовании, уменьшении затрат на процессы подтверждения подлинности.

2.4 Требования к системе

Продукт должен позволять совершать авторизацию пользователю по логину и паролю, давать доступ к приватному ключу и публичному адресу, получать документы для авторизованного пользователя и добавлять документы.

Use case’ы представлены на рис. 2.1.

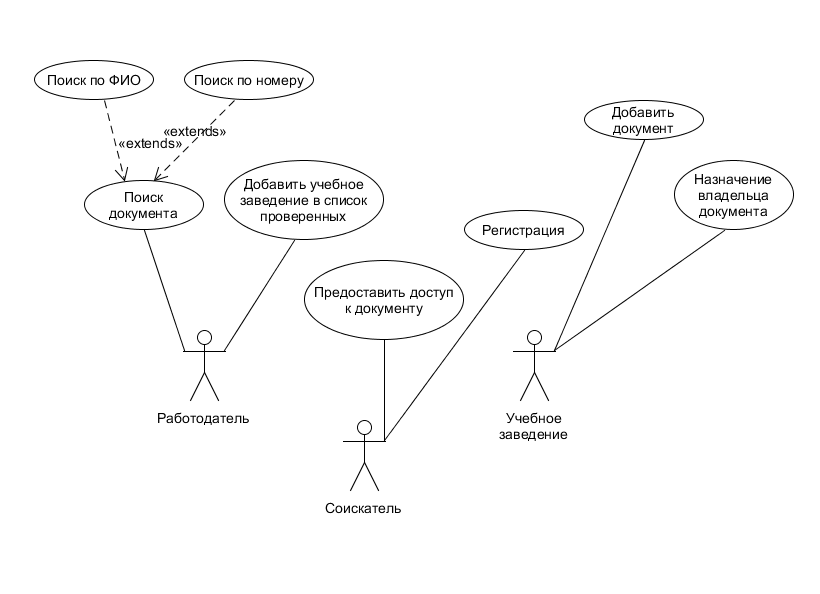


Рисунок 2.1 - Диаграмма прецедентов ИС Децентрализованный реестр документов об образовании

|  |  |  |  |
| --- | --- | --- | --- |
| Название функции | Входные параметры | Выходные параметры | Описание |
|  |  |  |  |

При разработке должны быть использованы следующие технологии и инструменты:

* платформа Ethereum – виртуальная машина для разработки децентрализованных приложений на базе Blockchain.
* Solidity – язык виртуальной машины Ethereum, используется для написания smart contracts
* TestRPS – эмулятор JSON RPC API Ethereum
* библиотека Web3j для Android – позволяет работать с JSON RPC API через обертки, без непосредственного ручного составления запросов
* Remix – онлайн компилятор Solidity. Преобразует контракт на языке Solidity в метаданные и байт-код виртуальной машины Ethereum.
* web3j wrapper – инструмент для работы со смарт контрактами на языке Java
* Android Studio 3.0
* NinjaMock – инструмент проектирования пользовательского интерфейса

## 2 Проектирование серверной части

Серверная часть системы должна состоять из хранилища (блокчейн), смарт контрактов и внешнего API, позволяющего взаимодействовать с другими модулями системы.

Реализация технологии блокчейн предоставляется платформой Ethereum, как и JSON RPC API. Контракты должны быть разработаны самостоятельно. Контракты должны определять структуру хранимых в блокчейне данных и методы взаимодействия с ними извне.

В блокчейне должны храниться данные о пользователях системы и документах, которые были добавлены пользователями. При этом наружу должен предоставляться API для взаимодействия внешнего модуля и хранилища.

В таблице 2.1 представлены методы серверной части и их параметры.TODO

Таблица 2.1 - API серверной части

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Выходные параметры | Описание |
| addDocument | string data (JSON),  string fio | - | Добавление документа с параметрами, перечисленными в data в формате JSON и с ФИО владельца в параметре fio |
| getDocById | uint id | uint id,  address owner,  string data,  string fio | Возвращает параметры документа по его номеру |
| getDocumentsNumber | - | uint number | Возвращает количество документов в хранилище |
| getThisAddresDocNumber | - | uint number | Возвращает количество документов, успешно добавленных текущим пользователем |
| getThisAddresDocById | uint id | uint id,  address owner,  string data,  string fio | Возвращает параметры документа текущего пользователя по номеру документа в списке |

В табл. 2.1 входные и выходные параметры приведены в терминологии языка Solidity. Значения некоторых из них:

* uint – тип данных беззнакового целого числа ненормированной длины
* string – тип данных динамически расширяемой строки в кодировке UTF-8
* address – тип данных, определяющий ссылки в хранилище блокчейна, представлен 20-байтовым значением

На рис. 2.1 представлена диаграмма, отражающая действие смарт контрактов, написанных на языке Solidity, на виртуальную машину Ethereum.

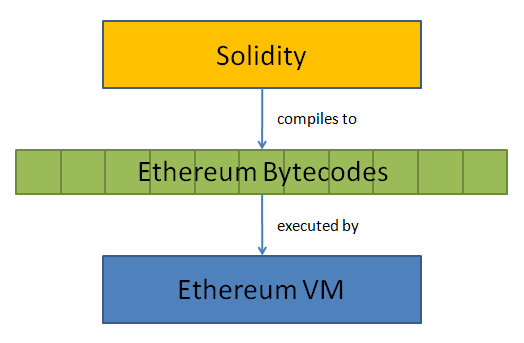


Рисунок 2.1. Взаимодействие смарт контрактов и Ethereum VM

Подготовка серверной части завершается развертыванием контракта(ов) в блокчейне и сохранением его адреса.

## 3 Проектирование клиентской части

JSON RPC API, предоставляемый виртуальной машиной Ethereum, требует достаточно много трудозатрат для использования, так как предполагает отправку и получение данных в виде 16-ричных строк. Это не позволяет разработчику вручную составлять запросы к Ethereum VM. К счастью, Ethereum – достаточно развитая платформа, поэтому есть набор инструментов, облегчающих доступ к данным в блокчейне из клиентских приложений.

На рис. 2.2 представлена диаграмма развертывания системы.

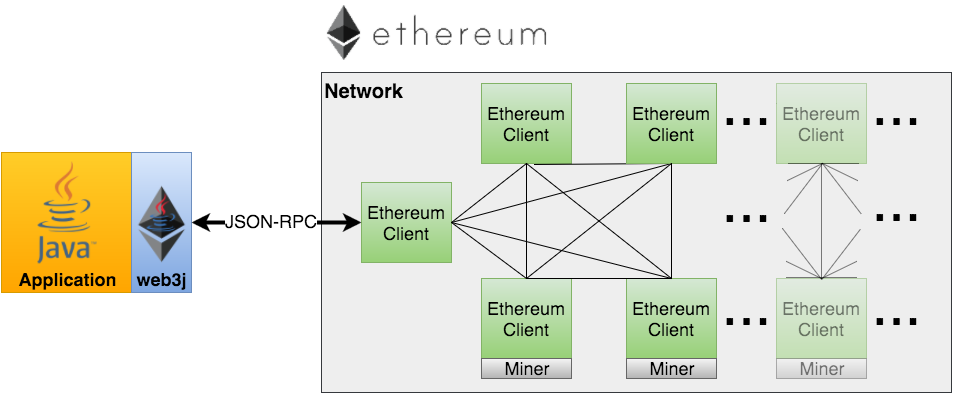


Рисунок 2.2. Архитектура децентрализованного приложения

Для данного продукта был выбран инструмент Web3j. Как указано на странице официальной странице, web3j – легкая Java и Android библиотека для интеграции с клиентом Ethereum (имеется в виду JSON RPC API).

Помимо библиотеки для Java web3j поставляет командные инструменты для некоторых других действий, не предназначенных для исполнения во время выполнения. Например, wrapper – утилита для создания класса-обертки для контракта на языке Java. Этот класс необходимо включить в проект для взаимодействия с внешним API клиента Ethereum.

Так как wrapper web3j использует метаданные и байт-код контракта, а не команды на языке Solidity, контракт необходимо сначала скомпилировать. Для этой цели удобно пользоваться инструментом Remix – онлайн компилятором Solidity.

Получение и отправка данных в Android – это задачи, требующие длительного времени, поэтому их нельзя выполнять в главном потоке. Для использования потоков необходимо использовать специальные инструменты, например, AsyncTask. Для каждого запроса необходимо написать свой класс AsyncTask.

Помимо перечисленных выше компонентов, необходимо разработать классы-помощники работы с специфическими данными, например, класс для работы с профилем пользователя на устройстве. Более подробно об этом рассказано в разделе «Реализация».

## 4 Проектирование пользовательского интерфейса

Для работы пользователю необходимы будут следующие формы:

* авторизация
* главная форма
* добавление документа

TODO

В связи с этим, были разработаны следующие макеты:

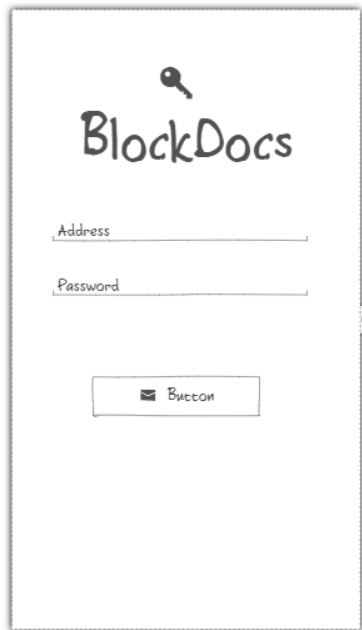


Рисунок 2.3. Макет формы авторизации

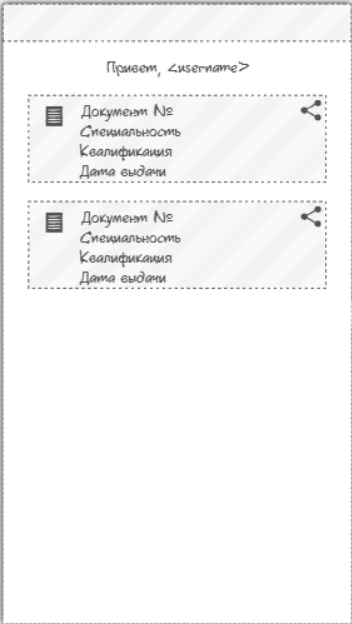


Рисунок 2.4. Макет главной формы

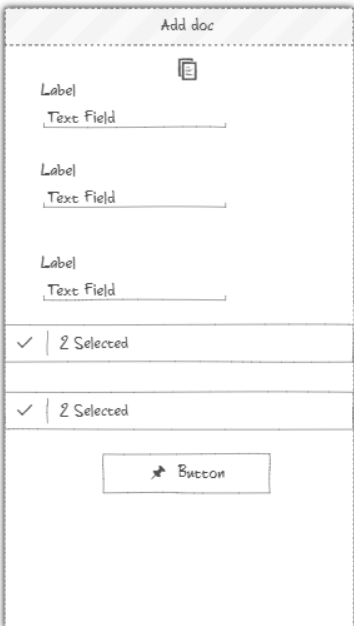


Рисунок 2.5. Макет формы добавления документа

Дизайн форм должен быть приближен к material design, что должно быть достигнуто использованием специфических компонентов.

6.1 Организационная структура

На рисунке 6.1 представлена организационная структура автоматизируемого объекта.

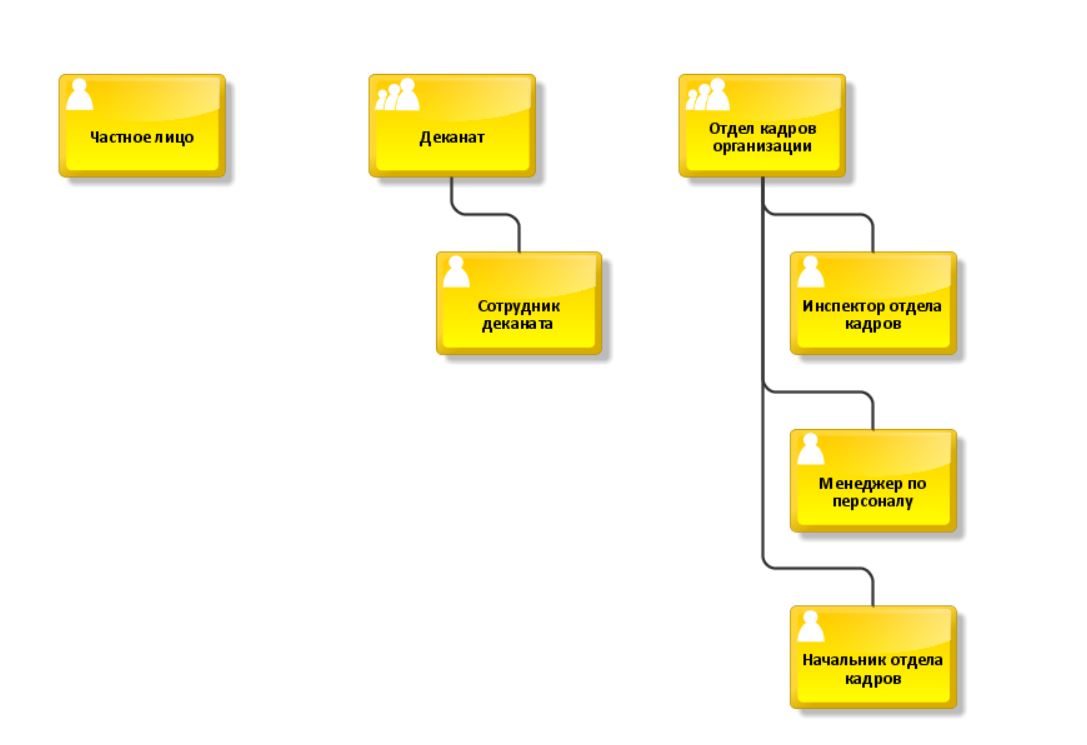


Рисунок 6.1 - Организационная структура

6.2 Информационное обеспечение

На рисунке 6.2 представлена схема информационного обеспечения.

Особенность данной информационной системы в том, что в качестве хранилища данных используется не база данных, как это принято в большинстве современных систем, а блокчейн. Блокчейн кроме всего прочего, что может содержаться в БД, сохраняет все сведенья об изменении данных в хранилище. Данные в блокчейне хранятся с помощью смарт-контрактов – одновременно и вместилищ информации, и спецификаций по их обработке. Контракт создается один раз и работает весь жизненный цикл системы.

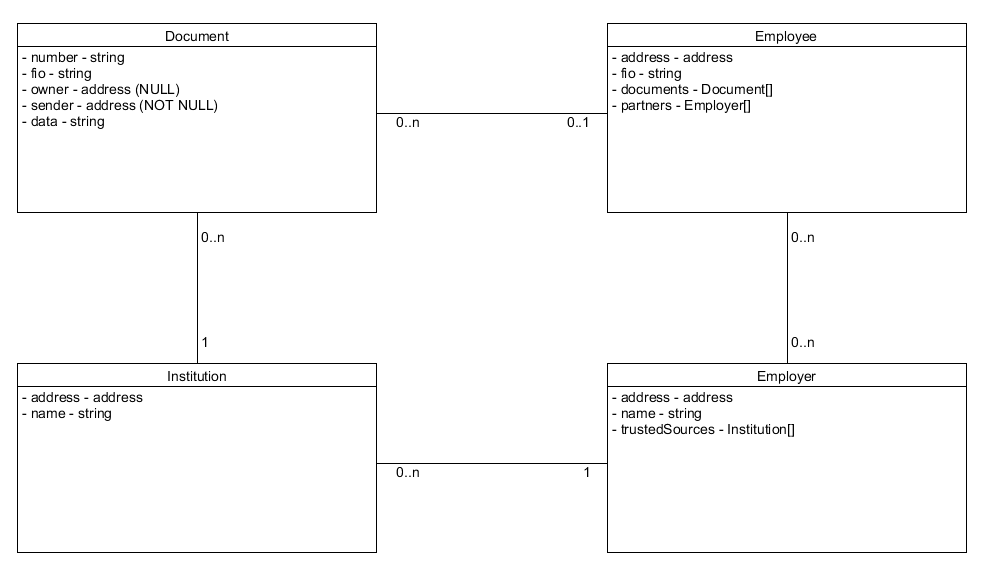


Рисунок 6.2 - Схема информационного обеспечения ИС Децентрализованный реестр документов об образовании

Блокчейн не содержит таблиц в чистом виде – в контракт можно загрузить любые данные, приведенные к строке или числу. Например, для хранения данных, по которым не предусмотрен поиск используется одно поле – data. Оно содержит сведенья о документе об образовании, представленные в формате JSON. Подразумевается, что речь о нормальных формах, применимых исключительно к реляционным БД, здесь вести нельзя.

6.3 Математическое обеспечение

Алгоритм работы приложения представлен на рисунке 6.3.



Рисунок 6.3 - Алгоритм учета документа

Наименование алгоритма: учет документа.

Назначение: предназначен для добавления документа в реестр, определения его владельца и указания ссылки на владельца в документе.

Входные данные: оригинал документа об образовании, сведенья о владельце, учетные данные владельца (для зарегистрированных).

Выходные данные: сведенья об успешности завершения процедуры.

Схема, моделирующая данные системы, представлена на рисунке 6.4.



Рисунок 6.4 - Схема данных

Схема взаимодействия программных модулей представлена на рисунке 6.5.



Рисунок 6.5 - Схема взаимодействия программных и технических модулей

6.4 Техническое обеспечение

Структурная схема ИС представлена на рисунке 6.5.



Рисунок 6.5 – Структурная схема ИС

СВИ – средство выхода в интернет

ВН – вычислительная нода

ИН – информационная нода

КБ – клиент блокчейна

ПИП – программный интерфейс приложения

W3J – библиотека web3j

МУ – мобильное устройство

СЭ – сенсорный экран

ХК – хранилище ключей

## 5 Разработка мероприятий по повышению надежности и достоверности выдаваемой информации

7.1 Повышение надежности



Рисунок 7.1 - Схема расчета надежности ИС

При решении поставленной задачи предполагается известным достигнутый уровень вероятности безотказной работы группы технических средств Pi(t)i = 1,2,...N и вводятся следующие допущения:

При решении любой задачи в ИС используются все технические средства системы;

Схема расчета надежности последовательная

Вероятность правильного решения задачи системой в заданном интервале времени РЗ(t) зависит только от правильной работы технических средств РЗ(t) = P(t);

Все задачи, решаемые системой, имеют одинаковую заданную вероятность правильного решения.

Тогда можем определить вероятность правильного решения задачи:

, i=1,2,…N (7.1)

Сенсорный экран – λ = 1,05\*10-4 1/час (первичные датчики)

Мобильное устройство - λ = 2,00\*10-4 1/час (персональная ЭВМ)

Сетевая карта – λ = 1,16\*10-4 1/час (распределительные устройства)

Канал связи – λ = 0,10\*10-4 1/час (линии связи)

Вычислительная нода – λ = 1,00\*10-4 1/час (процессор)

Время непрерывной работы – 8 часов.

Вероятность безотказной работы незарезервированной подсистемы равна: 0,986.

7.2 Повышение достоверности выдаваемой информации

Достоверность выдаваемой информации находится по формуле:

, (7.2)

где – достоверность выдаваемой информации;

Р – вероятность безотказной работы подсистемы;

Q – вероятность отказа подсистемы;

k – условная вероятность обнаружения ошибки;

h – коэффициент достоверности условной вероятности.

Исходные данные, рассчитанные ранее:

заданная достоверность: J = 0,99;

вероятность безотказной работы: Pзаданн(t)= 0,986

вероятность отказа подсистемы .

При известной вероятности исправной работы системы по формуле найдем:

При вероятности безотказной работы устройства контроля P\* = 0,80 вероятность отказа устройства контроля:

Рассчитаем коэффициент достоверности контрольной информации по формуле 6.

Из этого следует, что:

Для того чтобы подсистемы обеспечивала заданный уровень достоверности выдаваемой информации, подключать устройство контроля с k = 0,98.

# РЕАЛИЗАЦИЯ

## 1 Реализация серверной части

В качестве серверной части выступает виртуальная машина Ethereum с развернутым смарт контрактом. Структура контракта на Solidity представлена на рис.3.1.

В первой строке указывается текущая версия Solidity. Самая новая версия на момент написания работы – 4.19, но это не самая стабильная версия. Далее идет название контракта – Documents. Внутри описания контракта содержится описание структуры данных, которая используется внутри контракта. Область видимости структуры – текущий контракт. Структура Document – эквивалент документа об образовании. Он характеризуется адресом владельца, данными в формате JSON и ФИО. Данные, по которым осуществляется поиск внутри

контракта, необходимо вынести из строки данных. Остальные данные можно хранить в одной строке в формате JSON. Из-за особенностей языка Solidity, который не позволяет полноценно работать со строками, мы не можем хранить все данные в строке data. С другой стороны, Solidity не поддерживает возврат функциями объектов (экземпляров структур), но позволяет возвращать множественные параметры. Для того, чтобы не раздувать выходные параметры, данные объединены в строку.

Для хранения документов используется динамический массив.

Все методы делятся на транзакции – изменяющие состояние контракта, - и функции – считывающие состояние. В нашем случае добавление документа будет транзакцией, а все остальные методы – функциями.

После составления контракта его необходимо развернуть в блокчейне. Пока контракт развернут, все его данные хранятся распределенными в сети. У контракта тоже есть адрес, как и у пользователей, а значит, он может быть участником транзакций.

Это свойство используется для тестирования контрактов. 

Рисунок 3.1. Структура контракта

## 2 Тестирование контракта

Для тестирования контракта используется среда разработки Truffle. На рис. 3.2 представлена структура проекта Truffle.

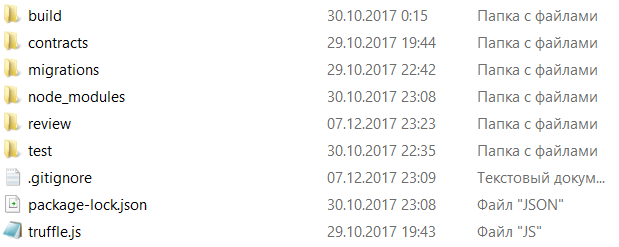


Рисунок 3.2. Структура проекта Truffle

Основная часть проекта – контракты – расположена в папке contracts. В папке build располагаются скомпилированные файлы контрактов. Папка test содержит тесты на языке Solidity и javascript.

Для тестирования контракта следует написать другой контракт, поместить его в папку test и запустить командой truffle test (рис.3.3).

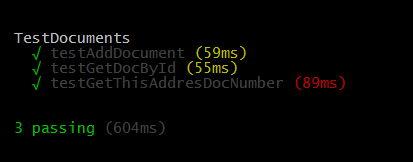


Рисунок 3.3. Тестирование контрактов

Так как контракт имеет собственный адрес, он может быть отправителем транзакций, а в нашем случае – владельцем документов.

## 3 Реализация клиентской части

В качестве клиента выступает приложение под платформу Android. Существует два типа клиентов для сети блокчейн – толстый клиент, который содержит копию хранилища, и тонкий клиент, который работает с хранилищем с помощью запроса.

Требование разработать приложение для мобильной платформы накладывает некоторые ограничения на продукт, а именно - оно должно занимать как можно меньше места на устройстве. В связи с этим сохранение полной копии хранилища на устройстве не представляется возможным.

На рис. 3.4. представлена структура проекта Android-приложения.

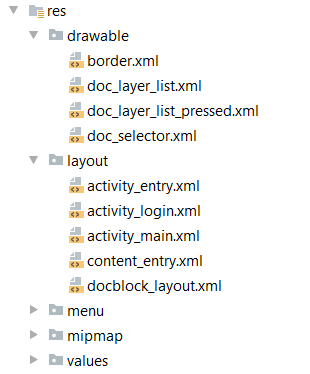
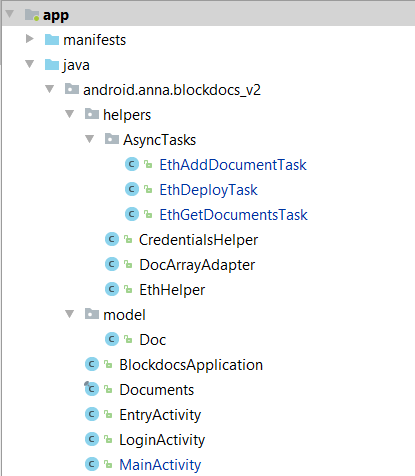


Рисунок 3.4. Структура проекта Android-приложения

* В качестве модели используется класс Doc, экземпляр которого содержит данные о конкретном документе.
* DocArrayAdapter – класс, который помогает правильно заполнить ListView на главной форме.
* CredintailsHelper – класс для безопасной работы с файлом кошелька на устройстве
* EthHelper – слой, реализующий доступ к хранилищу с помощью запросов
* AsyncTasks – набор однотипных классов для выполнения различных запросов в отдельном потоке
* Documents – класс-обертка для работы со смарт-контрактом на языке Java
* BlockDocsApplication – кастомная реализация класса приложения для хранения необходимых констант
* набор классов Activity – классы для обработки событий на формах

Основной особенностью данного приложения является использование библиотеки web3j и класса-обертки смарт-контракта. Класс-обертка генерируется инструментами командной строки от web3j и содержит следующие методы (рис. 3.5).

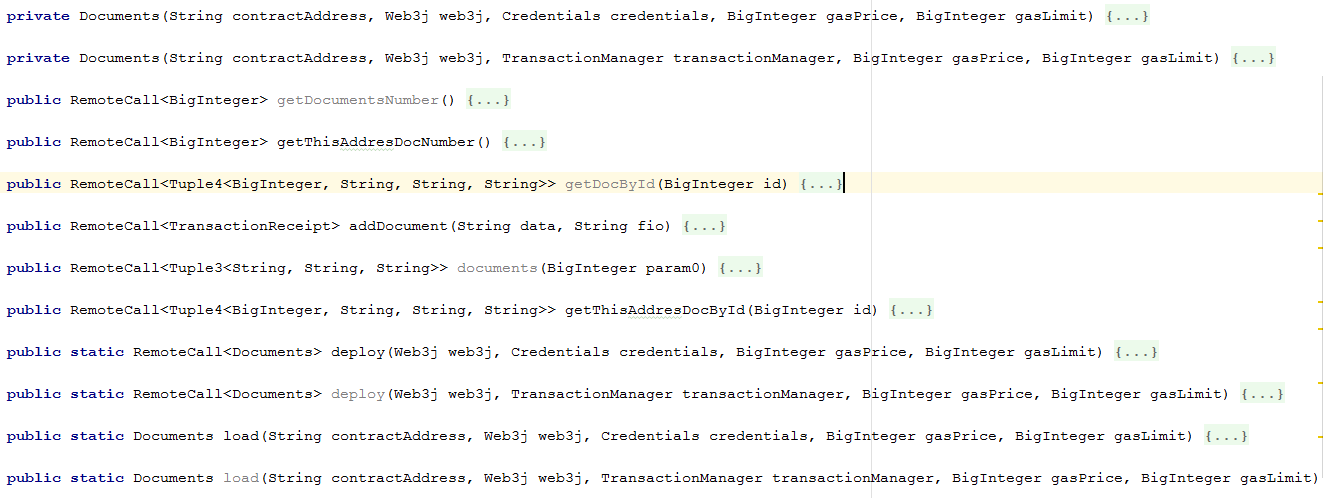


Рисунок 3.5 - Методы класса Documents

Методы, за исключением статических, совпадают с методами контракта и отличаются только типом выходных параметров. Проблема заключается в следующем: Solidity поддерживает возврат нескольких значений, а Java нет. Для решения этой проблемы web3j поставляет набор классов под названием Tuple («кортеж»). Например, Tuple2<String, String> содержит в себе ровно два возвращаемых контрактом значения.

Статические методы нужны для:

* развертывания контракта в блокчейне с помощью мобильного клиента (эта функция не используется в приложении, так как контракт уже развернут раз и навсегда в сети блокчейн)
* выгрузки контракта по его адресу – таким образом можно получить экземпляр контракта для дальнейшей работы с ним.

## 4 Тестирование клиентской части

Для тестирования клиентской части используется эмулятор сети блокчейн с готовыми аккаунтами – TestRPC. Он предоставляет JSON RPC API, с помощью которого можно развернуть, выгрузить контракт, создавать транзакции, вызывать функции контракта и т.п. На рис. 3.6. представлен пользовательский интерфейс TestRPC.



Рисунок 3.6. Интерфейс эмулятора TestRPC

Эмулятор запущен в локальной сети, поэтому адрес и используемые аккаунты прописаны в коде.

Дальнейшее тестирование состоит в проверке на соответствие продукта функциональным требованиям.

# ИНФОРМАЦИОННЫЙ МЕНЕДЖМЕНТ

В сфере информационного менеджмента рассматриваются процессы управления проектом на всех этапах его жизненного цикла. При этом в информационный менеджмент в широком смысле занимается задачами, связанными не только с данными, но и со всеми другими ресурсами, которые косвенно или напрямую взаимодействуют с информацией.

В данной главе рассматриваются следующие задачи ИМ:

1. Управление капиталовложениями – эта задача решается путем расчета полной стоимости владения ИС ДРДО, включая затраты на создание и сопровождение.
2. Управление персоналом – составление должностной инструкции по работе с ИС ДРДО.
3. Развитие информационной системы и обеспечение ее обслуживания – проведение SWOT-анализа ИС ДРДО с привлечением экспертов в данной области.

## 3 SWOT-анализ ИС ДРДО

SWOT-анализ – это метод стратегического планирования, который заключается в выявлении сильных и слабый сторон информационной системы, а также в определении влияния, которые эти стороны оказывают на потенциальные возможности и угрозы для рассматриваемой системы. SWOT-анализ помогает решить одну из задач информационного менеджмента, а именно – «Развитие системы и обеспечение ее обслуживания».

Этап 1. Для выявления необходимых для анализа факторов был проведен подбор группы экспертов. При выборе экспертов была учтена психологическая совместимость (коэффициент совместимости по группе – 0,94 > 0,6).

Участники группы экспертов:

1. Горева А.Д. – исполнитель проектных работ
2. Вершинин В.В. – руководитель проекта
3. Горев А.П. – ведущий разработчик BV LedgerLeopard, специалист по смарт-контрактам

Эксперты предложили набор сильных, слабых сторон, возможностей и угроз ИС ДРДО. При этом коэффициент конкордации составил более 0,75, что говорит о достаточной согласованности экспертов в этом вопросе.

Экспертами сформулированы следующие сильные и слабые стороны, а также возможности и угрозы:

Сильные стороны:

* S1 – безопасность хранения данных
* S2 – гарантированная подлинность документов
* S3 – высокая скорость доступа к данным
* S4 – предупреждение коррупции

Слабые стороны:

* W1 - недостаточная известность
* W2 - дефицит финансовых ресурсов
* W3 - низкая мотивация персонала
* W4 – проблемы с масштабируемостью

Возможности:

* O1 – упразднение архивов
* O2 – расширение типов документов, возможность добавления документов, не связанных с образованием
* O3 – выход на федеральный уровень
* O4 – выход на мировой уровень

Угрозы:

* T1 – проблема 51-го процента
* T2 – человеческий фактор
* T3 – компрометирование технологии в будущем (возможно)
* T4 – потеря текущей команды разработки

Этап 2. Формирование матрицы SWOT-анализа.

Матрица строится по следующему принципу - экспертная группа указывает следующие показатели:

* Pj – вероятность появления угрозы или возможности для рассматриваемой информационной системы. Может принимать значения от 0 до 1 и нечетко делится на Низкую (0 – 20%), Среднюю (20 – 50%), Высокую (50 – 85%) и Наиболее вероятную (85 – 100%).
* Kj – коэффициент влияния угрозы или возможности на дальнейшую деятельность предприятия. Так же оценивается значениями от 0 до 1 по шкале Нет влияния (0%), Низкое влияние (1 – 25%), Достаточное влияние (25 – 50%), Серьезное влияние (50 – 90%), Кардинальные изменения в процессах компании (90 – 100%).
* Ai – интенсивность сильных сторон оценивается от 1 до 5 баллов включительно. Интенсивность слабых сторон оценивается от -1 до -5 (-1 – наименьшая интенсивность, -5 – наибольшая) включительно. Интенсивность определяет, насколько значительное преимущество получает предприятие из-за своей сильной стороны и насколько сильные потери получит из-за слабых сторон.

Таблица 1 - Матрица SWOT-анализа

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Интенсивность (Аi) | Возможности (О) | |  |  | Угрозы (Т) | |  |  |
|  |  | О1 | О2 | О3 | O4 | Т1 | Т2 | Т3 | T4 |
| Вероятность появления (Pj) | | 0,5 | 0,9 | 0,7 | 0,2 | 0,1 | 0,8 | 0,1 | 0,2 |
| Коэффициент влияния (Кj) | | 0,9 | 0,5 | 1 | 1 | 1 | 0,4 | 1 | 0,5 |
| S1 | 3 | 4 | 3 | 3 | 2 | 2 | 5 | 2 | 1 |
| S2 | 5 | 5 | 4 | 5 | 4 | 2 | 5 | 1 | 1 |
| S3 | 2 | 4 | 2 | 3 | 3 | 1 | 4 | 1 | 1 |
| S4 | 5 | 2 | 4 | 5 | 4 | 3 | 5 | 1 | 2 |
| W1 | -5 | 5 | 5 | 5 | 5 | 4 | 1 | 1 | 4 |
| W2 | -3 | 2 | 3 | 2 | 2 | 3 | 4 | 1 | 5 |
| W3 | -2 | 2 | 4 | 2 | 2 | 1 | 5 | 1 | 4 |
| W4 | -4 | 4 | 5 | 5 | 5 | 4 | 1 | 1 | 2 |

В ячейках аij указывается способность сильных сторон содействовать реализации возможностей и противостоять угрозам и способность слабых сторон ослабить воздействие возможностей и усилить угрозы. Для упрощения процесса оценки рекомендуется использовать следующую шкалу:

Оценки в этих квадрантах должны выставляться без учета реальной интенсивности фактора для организации, т.к. это уже учтено в столбце интенсивность (Аj), то есть проводятся экспертные оценки влияния силы или слабости номинального выявленного фактора на отмеченные возможности или угрозы.

Этап 3 – Преобразование матрицы.

Преобразование исходной матрицы осуществляется на основании следующей формулы:

Aij= Ai\*Kj\*Pj\*aij

Затем производится суммирование полученных оценок по строкам и столбцам матрицы, а также разработка выводов и рекомендаций.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Интенсивность (Аi) | Возможности (О) | |  |  | Угрозы (Т) | |  |  |
|  |  | О1 | О2 | О3 | O4 | Т1 | Т2 | Т3 | T4 |
| Вероятность появления (Pj) | | 0,5 | 0,9 | 0,7 | 0,2 | 0,1 | 0,8 | 0,1 | 0,2 |
| Коэффициент влияния (Кj) | | 0,9 | 0,5 | 1 | 1 | 1 | 0,4 | 1 | 0,5 |
| S1 | 3 | 5,4 | 4,05 | 6,3 | 1,2 | 0,6 | 4,8 | 0,6 | 0,3 |
| S2 | 5 | 11,25 | 9 | 17,5 | 4 | 1 | 8 | 0,5 | 0,5 |
| S3 | 2 | 3,6 | 1,8 | 4,2 | 1,2 | 0,2 | 2,56 | 0,2 | 0,2 |
| S4 | 5 | 4,5 | 9 | 17,5 | 4 | 1,5 | 8 | 0,5 | 1 |
| W1 | -5 | -11,25 | -11,25 | -17,5 | -5 | -2 | -1,6 | -0,5 | -2 |
| W2 | -3 | -2,7 | -4,05 | -4,2 | -1,2 | -0,9 | -3,84 | -0,3 | -1,5 |
| W3 | -2 | -1,8 | -3,6 | -2,8 | -0,8 | -0,2 | -3,2 | -0,2 | -0,8 |
| W4 | -4 | -7,2 | -9 | -14 | -4 | -1,6 | -1,28 | -0,4 | -0,8 |

Вывод:

1. Наиболее сильные стороны: S2 – гарантированная подлинность документов и S4 – предупреждение коррупции.
2. Наиболее важными возможностями системы, воспользоваться которыми можно при помощи сильных сторон, являются:

* O3 – выход на федеральный уровень
* O1 – упразднение архивов

При этом они же являются наиболее уязвимыми из-за слабых сторон W1 (недостаточная известность) и W4 (проблемы с масштабируемостью).

1. Самая слабая сторона - W1 - недостаточная известность, она создает значительные препятствия при достижении почти всех перечисленных возможностей.
2. Выявлено, что угрозы маловероятны и успешно решаются сильными сторонами S2 (гарантированная подлинность документов) и S4 (предупреждение коррупции), к тому же, они не усугубляются слабыми сторонами.