МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Благовещенский государственный педагогический университет»

Физико-математический факультет

Кафедра информатики и методики преподавания информатики

Допустить к защите

Зав. кафедрой

\_\_\_\_\_\_\_\_\_\_\_\_\_ Л.А. Десятириктва

«\_\_\_\_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 г.

Автоматизированное веб-приложение с использованием технологии блокчейн

Выпускная квалификационная бакалаврская работа по направлению подготовки 02.03.03 – Математическое обеспечение и администрирование информационных систем,  
профиль «Технология программирования»

|  |  |  |  |
| --- | --- | --- | --- |
| Исполнитель:  студент группы 4А | ­­­­­­­­­­­­­­­­  *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ дата* | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись* | С.А. Либединский |
| Руководитель:  преподаватель | ­­­­­­­­­­­­­­­­  *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ дата* | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись* | М.А. Серов |
| Рецензент:  Преподаватель | ­­­­­­­­­­­­­­­­  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *дата* | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись* | М.А. Серов |
| Нормоконтроль:  Преподаватель | ­­­­­­­­­­­­­­­­  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *дата* | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись* | М.А. Серов |

Защита состоялась «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_2018 г. Оценка «\_\_\_\_\_\_\_\_\_\_»

|  |  |  |  |
| --- | --- | --- | --- |
| Председатель ГАК:  преподаватель | ­­­­­­­­­­­­­­­­  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *дата* | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись* | М.А. Серов |

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ](#_Toc445457375) 4

[1 Описание использованных средств разработки и теории](#_Toc445457375) 6

[1.1 Блокчейн 6](#_Toc445457377)

[1.2 Майнинг 6](#_Toc445457377)

[1.3 Описание использованных средств разработки 7](#_Toc445457377)

[1.3.1 Язык программирования java 7](#_Toc445457377)

[1.3.2 Spring 8](#_Toc445457378)

1.3.3 [JSON 9](#_Toc445457379)

1.3.4 [API «Telegram» 10](#_Toc445457379)

[1.3.5 Freemarker 10](#_Toc445457378)

[1.3.6 Tomcat 11](#_Toc445457378)

1.3.7 [Tyrus 1](#_Toc445457379)1

[2 Разработка приложений 0](#_Toc445457375)

[2.1 Общая структура приложений](#_Toc445457377) 0

[2.2 Использование json моделей](#_Toc445457377) 0

[2.3 Веб-приложение](#_Toc445457377) 0

[2.3.1 Структура веб-приложения](#_Toc445457377) 0

[2.3.2 Функции веб-приложения](#_Toc445457377) 0

[2.3.4 Использованные технологии в веб-приложении](#_Toc445457377) 0

[2.3.5 База данных веб-приложения](#_Toc445457377) 0

[2.3.6 Описание использования MVC в веб-приложении](#_Toc445457377) 0

[2.3.7 Взаимодействие веб-приложения с промежуточным обрабаывающим сервером](#_Toc445457377) 0

[2.3.8 Конфигурации веб-преложения](#_Toc445457377) 0

[2.3.9 Безопасность и авторизация веб-приложения](#_Toc445457377) 0

[2.3.10 Фронтенд и дизайн веб-приложения](#_Toc445457377) 0

[2.4 Промежуточный обрабатывающий сервер](#_Toc445457377) 0

[2.4.1 Структура промежуточно обрабатывающего сервера](#_Toc445457377) 0

[2.4.2 Функции промежуточно обрабатывающего сервера](#_Toc445457377) 0

[2.4.3 Конфигурация промежуточно обрабатывающего сервера](#_Toc445457377) 0

[2.4.4 Команды промежуточно обрабатывающего сервера](#_Toc445457377) 0

[2.4.5 Взаимодействие сервера с веб-приложением](#_Toc445457377) 0

[2.4.6 Взаимодействие сервера с клиентским приложением](#_Toc445457377) 0

[2.4.7 Взаимодействие сервере с прочими публичными интернет ресурсами](#_Toc445457377) 0

[2.4.8 Чат-бот «Телеграм»](#_Toc445457377) 0

[2.4.8.1 Описание взаимодействия чат-бота и сервера](#_Toc445457377) 0

[2.4.8.2 Регистрация клиента в сервисе «Telegram»](#_Toc445457377) 0

[2.4.8.3 Авторизация в сервисе «Telegram»](#_Toc445457377) 0

[2.4.8.4 Получение сообщений](#_Toc445457377) 0

[2.4.8.5 Отправка сообщений](#_Toc445457377) 0

[2.4.8 Функции чат-бота «Телеграм»](#_Toc445457377) 0

[2.4.8.1 Конфигурационный файл чат-бота](#_Toc445457377) 0

[2.4.8.2 Основные команды чат-бота](#_Toc445457377) 0

[2.4.8.3 Авторизация в чат-боте](#_Toc445457377) 0

[2.4.8.4 Описание файла конфигурации пользователей](#_Toc445457377) 0

[2.5 Клиентсое приложение на rig](#_Toc445457377) 0

[2.5.1 Структура локального приложения](#_Toc445457377) 0

[2.5.2 Функции локального приложения](#_Toc445457377) 0

[2.5.3 Файл хранения данных блокчейна](#_Toc445457377) 0

[2.5.4 Конфигурационный файл локального приложения](#_Toc445457377) 0

[2.5.5 Взаимодействие клиентского приложения с промежуточным сервером](#_Toc445457377) 0

[ЗАКЛЮЧЕНИЕ](#_Toc445457377) 0

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ](#_Toc445457377) 0

[ПРИЛОЖЕНИЯ](#_Toc445457375) 0

ВВЕДЕНИЕ

В 21 веке люди придумали разновидность [цифровой валюты](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D0%B2%D0%B0%D0%BB%D1%8E%D1%82%D0%B0), создание и контроль за которой базируются на [криптографических](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F) методах (Криптовалюта). Криптовалюта, появляется двумя способами, либо воспроизводится обычными пользователями системы или разработчиками криптовалюты. Если криптовалюту воспроизводят пользователи, то делают они это чаще всего с помощью специально собранных систем (rig), которые решают специфическую задачу, при решении которой выплачивают вознаграждение в виде криптовалюты – это и называется майнингом, то есть добываем цифровую валюту. Блокчейн же это распределенная база данных, которая хранит информацию обо всех транзакциях участников системы в виде «цепочки блоков».

Способы создание специальных систем добычи криптовалюты не очень сложны, но занимает много времени, сил и денег. Пользователям приходится проверять оборудование, как в программном, так и в техническом плане. Чаще всего из-за того что оборудование находится удаленно, проверить и контролировать что происходит с rig невозможно из-за удаленности. Чтобы упростить способ проверки программного состояния систем, а также избавится от услуг посторонних людей, можно разработать автоматизированную систему. Которая будет помогать контролировать бесперебойную работу устройств rig.

Целью данной работы стало разработки комплекса программного обеспечения приложений для удаленного управления и мониторинга над rig.

Объектом работы стало веб-приложение и чат-бот для мессенджера. А предметом работы послужил язык программирование, API(набор готовых [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [процедур](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [функций](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [структур](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) и [констант](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%81%D1%82%D0%B0%D0%BD%D1%82%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), предоставляемых приложением (библиотекой, сервисом) или операционной системой для использования во внешних программных продуктах) мессенджера и сторонние вспомогательные средства.

Для выполнения были поставлены следующие задачи:

* Разработка веб-приложения доя получения, отображения и отправки информации.
* Разработка промежуточного обрабатывающего сервера.
* Разработка клиентского приложения – rig;

1 ОПИСАНИЕ ИСПОЛЬЗОВАННЫХ СРЕДСТВ РАЗРАБОТКИ И ТЕОРИИ

* 1. Блокчейн

Блокчейн – распределенная база данных, которая хранит любую выбранную информацию обо всех транзакциях участников системы в виде «цепочки блоков». Доступ к реестру есть у абсолютно всех пользователей блокчейна, выступающих в качестве коллективного нотариуса, который подтверждает истинность информации в базе данных. Блокчейн может применяться для финансовых операций, идентификации пользователей, создания технологий [кибербезопасности](http://www.tadviser.ru/index.php/%D0%9A%D0%B8%D0%B1%D0%B5%D1%80%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D0%B8" \o "Кибербезопасности) и др.

Технология способна защитить данные, с которыми нам приходится работать, при этом сделав их более доступными и прозрачными. К тому же, блокчейн может заметно снизить затраты и минимизировать время, необходимое для решения возникающих проблем и устранения ошибок.

Представьте себе цифровую историю баланса виртуальных денежных средств: каждая запись и есть такой блок. У этой записи есть метка: дата и время внесения. Изначально считается обязательным запрет на изменение записей задним числом, потому что нужно, чтобы записи о балансе не допускали разных значений и оставались в исходном виде. К записям могут получить доступ только разработчик, у которого есть один закрытый ключ, и хозяин денежных средств, у которого есть другой. Затем к этой информации получат доступ только те, кому один из этих пользователей предоставит свой закрытый. Так, например, мы будем использовать блокчейн в нашей работе.

* 1. Майнинг

Майнинг, также добыча – деятельность по созданию новых структур (обычно речь идёт о новых блоках в [блокчейне](https://ru.wikipedia.org/wiki/%D0%91%D0%BB%D0%BE%D0%BA%D1%87%D0%B5%D0%B9%D0%BD" \o "Блокчейн)) для обеспечения функционирования [криптовалютных платформ](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B2%D0%B0%D0%BB%D1%8E%D1%82%D0%B0" \o "Криптовалюта). За создание очередной структурной единицы обычно предусмотрено вознаграждение за счёт новых (эмитированных) единиц криптовалюты и/или комиссионных сборов. Обычно майнинг сводится к серии вычислений с перебором параметров для нахождения [хеша](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%81%D1%83%D0%BC%D0%BC%D0%B0" \o "Хеш-сумма) с заданными свойствами. Разные криптовалюты используют разные модели вычислений, но они всегда достаточно длительны по времени для нахождения приемлемого варианта и быстры для проверки найденного решения. Для вычислений нужно любое устройства способное вычислять с выходом в сеть интернет, вплоть до калькуляторов.

В майнинге обычно подразумевается, что устройства будут работать, круглы сутки для увеличения прибыли.

В нашей работе потенциальные клиенты является майнеры (люди которые занимаются майнингом). С помощью нашей системы они смогут упростить способ майнинга.

* 1. Описание использованных средств разработки

1.3.1 Язык программирования java

Для решения выбора языка программирование java повлияло несколько факторов:

* Кроссплатформенность – способность программного обеспечения работать более чем на одной операционной системе;
* Многопоточность – распараллеливания потоков;
* Большое сообщество программистов ‒ большая поддержка языка и огромное количество документации;
* Объектно-ориентированный язык – это позволяет создавать модульные программы, исходный код, который может использоваться многократно.

Java – сильно типизированный объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle). Приложения Java обычно транслируются в специальный байт-код, поэтому они могут работать на любой компьютерной архитектуре, с помощью виртуальной Java – машины.

Java представляет собой язык программирования и платформу вычислений, которая была впервые выпущена Sun Microsystems в 1995 г. Существует множество приложений и веб-сайтов, которые не работают при отсутствии установленной Java, и с каждым днем число таких веб-сайтов и приложений увеличивается. Java отличается быстротой, высоким уровнем защиты и надежностью. От портативных компьютеров до центров данных, от игровых консолей до суперкомпьютеров, используемых для научных разработок, от сотовых телефонов до сети Интернет – Java повсюду!

1.3.2 Spring

Перед выбором технологии, на которой будет основываться архитектура, было выбрано Spring – как одна из самых популярных и поддерживаемых.

Spring Framework (или коротко Spring) – универсальный [фреймворк](https://ru.wikipedia.org/wiki/%D0%A4%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA" \o "Фреймворк) [с открытым исходным кодом](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) для [Java](https://ru.wikipedia.org/wiki/Java" \o "Java)-платформы. Также существует [форк](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BA" \o "Форк) для платформы [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework), названный Spring.NET.

Spring обеспечивает решения многих задач, с которыми сталкиваются Java-разработчики и организации, которые хотят создать информационную систему, основанную на платформе [Java](https://ru.wikipedia.org/wiki/Java" \o "Java). Из-за широкой функциональности трудно определить наиболее значимые структурные элементы, из которых он состоит. Spring не всецело связан с платформой [Java Enterprise](https://ru.wikipedia.org/wiki/J2EE" \o "J2EE), несмотря на его масштабную интеграцию с ней, что является важной причиной его популярности.

Фреймворк Spring MVC обеспечивает архитектуру паттерна Model–View–Controller при помощи слабо связанных готовых компонентов. Паттерн MVC разделяет аспекты приложения (логику ввода, бизнес-логику и логику UI), обеспечивая при этом свободную связь между ними.

Model (Модель) инкапсулирует (объединяет) данные приложения, в целом они будут состоять из POJO («Старых добрых Java–объектов», или бинов).

View (Отображение, Вид) отвечает за отображение данных Модели, – как правило, генерируя HTML, которые мы видим в своём браузере.

Controller (Контроллер) обрабатывает запрос пользователя, создаёт соответствующую Модель и передаёт её для отображения в Вид.

Вся логика работы Spring MVC построена вокруг DispatcherServlet, который принимает и обрабатывает все HTTP-запросы (из UI) и ответы на них.

Spring предоставляет три типа класса шаблонов для работы с JDBC:

* JdbcTemplate – (этот способ мы использовали в нашей статье) - Это основной способ доступа к базе. SQL параметры передаются как индексы Безопасность;
* NamedParameterJdbcTemplate – параметры передаются как именованная пара значений в Map
* SimpleJdbcTemplate – он использует особенности, такие как autoboxing и Generic.

1.3.3 JSON

Для хранения и передачи информации API «Telegram» использует json что и послужило его использование и распаривание. Достоево JSON:

* Удобочитаемость кода;
* Простота создания объекта данных на стороне сервера;
* Простота обработки данных на стороне клиента;
* Простота расширения;
* Отладка и исправление ошибок;
* Безопасность.

JSON – текстовый формат обмена данными, основанный на JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми и распознается ПО.

Json Simple - представляет собой простой API для обработки JSON файлов.

Google GSON – представляет собой простой API оборачивающий JSON в классы и обратно.

1.3.4 API «Telegram»

API представляет из себя HTTP-интерфейс для работы с ботами в Telegram.

В нем есть следующие возможности:

* Авторизация
* Отправка запросов
* Получение обновлений
* Типы
* Методы

В данной работе использовалась специальная библиотека для java, которая оборачивает API «Telegram» в классы и методы для удобства обращения к серверам.

1.3.5 Freemarker

FreeMarker ­­– компилирующий обработчик [шаблонов](https://ru.wikipedia.org/wiki/%D0%A8%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD), написанный на [Java](https://ru.wikipedia.org/wiki/Java" \o "Java). Один из инструментов, позволяющих отделить логику и данные от представления в духе концепции [Model–view–controller](https://ru.wikipedia.org/wiki/Model-view-controller" \o "Model-view-controller). Используется преимущественно при разработке web-приложений с использованием Java–сервлетов, также может использоваться для вывода текста в других случаях: генерация [CSS](https://ru.wikipedia.org/wiki/CSS), исходного кода Java и т. д. В отличие от [JSP](https://ru.wikipedia.org/wiki/JavaServer_Pages) FreeMarker не является зависимым от архитектуры сервлета или от протокола [HTTP](https://ru.wikipedia.org/wiki/HTTP). Таким образом, шаблонизатор может использоваться не только в web-проектах.

1.3.6 Tomcat и Tyrus

Tomcat (в старых версиях – Catalina) – [контейнер сервлетов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%82%D0%B5%D0%B9%D0%BD%D0%B5%D1%80_%D1%81%D0%B5%D1%80%D0%B2%D0%BB%D0%B5%D1%82%D0%BE%D0%B2) с открытым исходным кодом, разрабатываемый [Apache Software Foundation](https://ru.wikipedia.org/wiki/Apache_Software_Foundation" \o "Apache Software Foundation). Реализует спецификацию [сервлетов](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%BB%D0%B5%D1%82" \o "Сервлет), спецификацию [JavaServer Pages](https://ru.wikipedia.org/wiki/JavaServer_Pages" \o "JavaServer Pages) (JSP) и [JavaServer Faces](https://ru.wikipedia.org/wiki/JavaServer_Faces" \o "JavaServer Faces) (JSF). Разработан на языке [Java](https://ru.wikipedia.org/wiki/Java" \o "Java).

Tomcat позволяет запускать [веб-приложения](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5), содержит ряд программ для самоконфигурирования.

Tyrus – является открытым исходным кодом [JSR 356 – Java API, для](https://projects.eclipse.org/projects/ee4j.websocket) эталонной реализации [WebSocket](https://projects.eclipse.org/projects/ee4j.websocket) и легкой разработки приложений с использованием WebSocker. Протокол WebSocket, определенный IETF, обеспечивает двунаправленную связь между сервером и удаленным хостом. Преимуществом в основном является способность общаться в обоих направлениях, низкая латентность и небольшая ресурсоемкость. Поэтому Tyrus и WebSocket в чаще всего подходят для веб-приложений, которые требуют отправки огромного объема относительно небольших сообщений, таких как онлайн-игры или часто посещаемых веб-приложений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Java™ Platform, Standard Edition 8 API Specification: [сайт]. URL: https://docs.oracle.com/javase/8/docs/api/

2. Javadoc for JSON simple: [сайт]. URL: <http://alex-public-doc.s3.amazonaws.com/json_simple-1.1>

3. Telegram Bot API: [сайт]. URL: <https://core.telegram.org/bots/api>

4. Telegram Bot Java Library: [сайт]. URL: <https://github.com/rubenlagus/> TelegramBots

5. Learn Spring: [сайт]. URL: https://spring.io/docs