Министерство науки и высшего образования Российской Федерации

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

Институт компьютерных технологий и информационной безопасности

Кафедра информационно-аналитических систем безопасности им. Л.С. Берштейна

Структурное подразделение

Специальность/направление подготовки (шифр, название)

10.03.01 Информационная безопасность

**ОТЧЕТ**

О прохождении производственной практики

(преддипломная практика)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Фамилия | Белохвостов | | | | |
| Имя | Виктор | | | | |
| Отчество | Олегович | | | | |
| Обучающийся | *A picture containing chain, animal, mirror  Description automatically generated* | | | / | Белохвостов В.О. |
|  | подпись | | |  | расшифровка подписи |
| Место практики | | ООО «Дунайс Альянс» | | | |
|  | | наименование профильной организации | | | |
| Вид практики | | производственная практика | | | |
| Тип практики | | научно-исследовательская практика | | | |
| Способ проведения практики | | | стационарная (Таганрог) | | |
| Форма проведения практики | | | дискретная (по видам практик) | | |
| Сроки прохождения практики | | | с «15» апреля 2020 г. по «31» мая 2020 г. | | |

|  |  |  |
| --- | --- | --- |
| Руководитель практики  от структурного подразделения ЮФУ  Фирсов М. И. |  | Руководитель практики  от Профильной организации  Дегтярёв А.С. |
| ФИО, подпись | ФИО, подпись, М.П. |

Таганрог

2020 г.

**Оглавление**

[Введение 3](#_Toc42978803)

[Цели и задачи работы 4](#_Toc42978804)

[1. Методика проведения исследований 5](#_Toc42978805)

[1.1 Анализ, выбор и изучение технологий современной web разработки 5](#_Toc42978806)

[1.2 Создание backend части приложения 7](#_Toc42978807)

[1.3 Начало создание frontend части приложения 12](#_Toc42978808)

[Вывод 14](#_Toc42978809)

[Список использованных источников 15](#_Toc42978810)

# **ВВЕДЕНИЕ**

Важной деталью при проверке налоговыми органами проводимых сделок является определение взаимозависимости между контрагентами. Этот фактор тщательно проверяется с целью установление корректности сумм выплат, тем самым удостоверяясь в том, что они не были уменьшены или завышены, т.к. данные манипуляции с суммами позволяют контрагентам искусственно увеличить налогооблагаемую базу, тем самым получить больший налоговый вычет, или уменьшить ее, с целью снижения суммы налогового обложения.

На текущий момент большинство компаний и предприятий ведут документацию и отчетность, хранящуюся в открытых источниках, доступ к которым открыт из любой точки мира, где есть подключение к глобально сети, при этом деятельность самой организации может целиком строиться в интернете. Именно поэтому направление web разработки является одной из самых актуальных тем на сегодняшний день.

В рамках данной практики были рассмотрены технологии и инструменты для построения web-приложений, а также применение полученных знаний и навыков для создания web-приложения, облегчающего анализ данных на взаимозависимость.

**ЦЕЛИ И ЗАДАЧИ РАБОТЫ**

**Цель работы:**

Целью проводимой работы является построение пользовательского web-приложения, основным функционалом которого является анализ данных компаний и их сотрудников на взаимозависимость.

**Задачи работы:**

1. Анализ и изучение современных инструментов web разработки и основных подходов к построению web-интерфейсов в глобальной сети интернет.
2. Практическое применение полученных знаний и подобранных инструментов для построения web-приложения для анализа данных на взаимозависимость.
3. **Методика проведения исследований**
   1. **Анализ, выбор и изучение технологий современной web разработки**

На сегодняшний день существует огромное количество всевозможных технологий и инструментов, предоставляющих возможность реализации web-приложения. После анализа всех возможны вариантов, как для backend стороны, где претендентами выступали языки PHP, C#, Java, а также связка языка программирования Python и фреймворка Django, так и для frontend стороны, где выбор состоял из нативного подхода, когда используется стандартный набор – HTML + CSS + JavaScript, и различных бибилиотек и фреймворков, например таких как jQuery или Angular, были выбраны следующие компоненты:

* для backend – **NodeJS** и фреймворк **Express**, а в качестве баз выбор пал на **MongoDB** - не реляционную БД, основная ответственность которой заключалась в хранении пользователей системы и основных сущностей, и на **Neo4j** – графовую БД, основными задачи которой были анализ и построение графов зависимостей между лицами анализируемого предприятия;
* для frontend – **ReactJS**, **Redux** и **Redux Saga** для запросов;

Исходя из выбранных технологий была изучена документация, необходимая для создания web приложения, и установлены на персональный компьютер для дальнейшей работы. После чего вручную с помощью установленной среды NodeJS были сгенерированы два проекта для backend и frontend частей соответственно. Вся разработка велась с использованием системы контроля версий Github, позволяющей оперативно отслеживать изменения в проекте, а также сохранять актуальную версию от возможных локальных повреждений (рисунок 1).

**A screenshot of a cell phone

Description automatically generated**

Рисунок 1. Репозиторий с проектом

* 1. **Создание backend части приложения**

Следующим шагом после генерации приложения с помощью NodeJS была установка всех пакетов, необходимых для реализации функционала заложенного на этапе планирования. Первым и одним из основных являлся фреймворк Express – минималистичный и гибкий веб-фреймворк для приложений Node.js, предоставляющий обширный набор функций для web-приложений. С его помощью было описано web-API сервера, код которого представлен на рисунке 2. На текущем этапе серверная часть уже позволяет принимать, обрабатывать и давать ответ на поступающие запросы от клиентской части.

A screenshot of a cell phone

Description automatically generated

Рисунок 2. API серверной части

Следующими после Express были установлены два пакета, необходимых для успешного соединения и дальнейшей работы выбранных баз данных. В ряд этих пакетов вошли mongoose, облегчающий описание моделей MongoDB базы (рисунок 3) и дальнейшую работу с ними, а также пакет ogmneo предоставляющий возможность производить манипуляции с данными в БД Neo4j.

A screenshot of a social media post

Description automatically generated

Рисунок 3. Модель User, описанная с помощью mongoose

После успешного подключения всех необходимых пакетов, в число которых также вошли: xlsx для дальнейшего импорта данных сразу из excel таблиц, morgan – для логирования запросов и coockie-parser, дающий возможность для чтения куки из входящих запросов, была описание и настройка обработчика маршрутов для серверной части.

Основываясь на исследовании кода большинства опытных разработчиков, использующих связку NodeJS и Express, можно выделить два основных пути реализации обработчиков маршрутов, основные отличия затрагивают не столько кодовую базу, сколько распределение компонентов, необходимых для обработки маршрута. Один из них заключается в группировке компонентов по типу, то есть все файлы с маршрутами, контроллеры, являющиеся функциями-обработчиками запросов, поступающих на вышеупомянутые маршруты, и всевозможные запросы к базам данных, описывающихся в отдельных файлах, хранятся отдельно по группам.

Второй же путь, который был выбран для реализации данного приложения, придерживается группировки всех компонент по обрабатываемым сущностям (рисунок 4). В данном варианте работать с кодом намного удобнее, т.к. все, что используется для обработки конкретного маршрута хранится в одной папке и находится под рукой, за счет чего экономится время на разработку.

A screenshot of a cell phone

Description automatically generated

Рисунок 4. Структура группировки файлов по сущностям

Реализованная версия обработчика описывает несколько групп маршрутов для полноценной работы приложения. В их число входят группа, отвечающих за авторизацию – **auth/**, работу с данными пользователей – **users/**, работу с рабочими областями – **work-area/**, с помощью которых пользователь сможет анализировать сразу несколько предприятий на взаимозависимость лиц, а также группа отвечающая за работу непосредственно с данными этих предприятий – **analytical-elements/**.

Каждый маршрут имеет свой обработчик, вышеупомянутый контроллер, в котором описывается вся логика обработки данных из поступающего запроса, а также составление и отправка ответа на него. Пример одного из контроллеров реализованного приложения можно наблюдать на рисунке 5.

A screenshot of a social media post

Description automatically generated

Рисунок 5. Контроллер создания рабочей области пользователя

По завершению описания функционала сервера все запросы были протестированы с помощью программного обеспечения Postman, позволяющего проверить все описанные маршруты путем отправки http-запросов, тем самым эмитируя работу реальной клиентской части. Интерфейс Postman, а также пример выполнения одного из реализованных маршрутов в приложении можно увидеть на рисунке 6.

A screenshot of a social media post

Description automatically generated

Рисунок 6. Результат выполнения запроса на регистрацию пользователя

* 1. **Начало создание frontend части приложения**

Следующим шагом после генерации frontend приложения, а также завершения создания серверной части приложения, было подготовка основных инструментов, установка всех необходимых пакетов, а также обдумывание дизайна будущей клиентской части.

В число установленных пакетов вошел Redux – менеджер состояния приложения, позволяющий хранить и передавать данные приложения в необходимую нам компоненту, на которых и строится React приложение, напрямую, а не последовательно, как бы это было реализовано без данного пакета.

Следующим шагом было начало работы по реализации обдуманного дизайна, результаты которой представлены на рисунках 7 и 8. Весь функционал клиентской части разбивается на небольшие компоненты, которые соединяются, путем включения одних, описывающих визуальную составляющую или же конкретный функционал приложения, в других, более объемные компоненты, представляющие собой завершенные логические блоки, примером которых могут послужить целые страницы web-приложения.

A close up of a logo

Description automatically generated

Рисунок 7. Начало реализации меню сайта с логотипом

A screenshot of a computer

Description automatically generated

Рисунок 8. Начало реализация страницы логина и регистрации

Таким образом Следующим шагом в исследовании было завершение создания пользовательского интерфейса и полноценное тестирование приложения в «боевой» среде.

# **ВЫВОД**

В ходе практики были проанализирован современный процесс разработки web-интерфейсов, были подобраны актуальные на текущий момент инструменты для создания собственного приложения, а также реализована большая часть серверной логики и меньшая часть пользовательского web-приложения для исследования компаний и их сотрудников на взаимозависимость.

Следующим шагом в исследовании будет завершение создания пользовательского интерфейса и загрузка приложения в глобальную сеть для полноценного использования.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Документация NodeJS – Режим доступа: [Электронный ресурс] URL: https://nodejs.org/docs/latest-v9.x/api (дата обращения: 15.05.2020).
2. Документация MongoDB – Режим доступа: [Электронный ресурс] URL: https://docs.mongodb.com/manual/ (дата обращения: 17.05.2020).
3. Документация Neo4j – Режим доступа: [Электронный ресурс] URL: https://neo4j.com/developer/javascript/ (дата обращения: 17.05.2020).
4. Документация ReactJS – Режим доступа: [Электронный ресурс] URL: https://reactjs.org/docs/ (дата обращения: 20.05.2020).
5. Начало работы с Express – Metanit – Режим доступа: [Электронный ресурс] URL: https://metanit.com/web/nodejs/4.1.php (дата обращения: 20.05.2020).
6. How to create a React frontend and a Node/Express backend – FreeCodeCamp – Режим доступа: [Электронный ресурс] URL: https://www.freecodecamp.org/news/create-a-react-frontend-a-node-express-backend-and-connect-them-together-c5798926047c/ (дата обращения: 25.05.2020).