МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное АВТОНОМНОЕ образовательное учреждение высшего образования

«Национальный исследовательский ядерный университет «МИФИ»

**Димитровградский инженерно-технологический институт –**

филиал федерального государственного автономного образовательного учреждения высшего образования

«Национальный исследовательский ядерный университет «МИФИ»

**(ДИТИ НИЯУ МИФИ)**

**Специальность 09.02.07 Информационные системы и программирование**

**ОТЧЕТ**

**по производственной практике**

**(по профилю специальности)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование практики** | | ПП.02.01 Производственная практика | | |
| **Профессиональный модуль:** | | ПМ.02 Осуществление интеграции программных модулей | | |
| **студента** | | Волкова Алексея Викторовича | | |
| **группы** | | 331 | | |
| **Место проведения практики:** полное наименование организации/предприятия–базы практики | | Общество с ограниченной ответственностью «СимбирСофт» | | |
| Адрес местонахождения | | ул. Куйбышева, 226Г/1, Димитровград | | |
| **Дата прохождения практики** | с «8» июня 2023 г. по «28» июня 2023 г. | | | | |
| **Руководители практики** | |  | | |
| От организации/ предприятия базы практики | | | Власенко Олег Федосович |
| От техникума  ДИТИ НИЯУ МИФИ | | | Надеждина Алёна Валерьевна |
| **Итог практики (оценка)** | | |  | | | |
| **Подпись руководителя практики от организации** | | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.Ф. Власенко  МП ИОФ | | | |
| **Подпись преподавателя** | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В. Надеждина | | |
| **Дата проверки** | | 28 июня 2023 г. | | |

Димитровград

**Содержание**

[ВВЕДЕНИЕ 3](#_Toc138800362)

[ПРАКТИЧЕСКАЯ ЧАСТЬ 5](#_Toc138800363)

[1 ЗАДАНИЕ №1 5](#_Toc138800364)

[2 ЗАДАНИЕ №2 11](#_Toc138800365)

[3 ЗАДАНИЕ №3 16](#_Toc138800366)

[4 ЗАДАНИЯ РАБОТАДАТЕЛЯ 23](#_Toc138800367)

[ЗАКЛЮЧЕНИЕ 35](#_Toc138800397)5

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 36](#_Toc138800398)6

# **ВВЕДЕНИЕ**

Производственная практика (по профилю специальности) является частью программы подготовки специалистов среднего звена (ППССЗ) в соответствии с ФГОС СПО по специальности 09.02.07 Информационные системы и программирование в части освоения квалификации «Программист» и основных видов профессиональной деятельности (ВПД).

Производственная практика (по профилю специальности) направлена на формирование у обучающегося общих и профессиональных компетенций, приобретение практического опыта и реализуется в рамках профессиональных модулей ППССЗ СПО по каждому из видов профессиональной деятельности, предусмотренных ФГОС СПО специальности.

Цель производственной практики является обобщение и совершенствование знаний, полученных студентами в процессе обучения, знакомство студентов с передовыми технологиями производства, автоматизацией на производстве и во вспомогательных службах, организацией труда, организацией процесса компьютерной обработки различных видов информации, приобретение умений организаторской работы по специальности.

Задачами производственной практики являются:

* развитие профессионального мышления и приобретение студентами профессиональных умений по специальности;
* приобретение практических умений и навыков техника - программиста по направлению (ПМ. 02);
* развитие и углубление навыков программирования;
* изучение и освоение программных систем, пакетов прикладных программ (ППП), фирменных программных продуктов;
* реализация функции программной защиты информации;
* закрепление, расширение и систематизация знаний, полученных при изучении профессиональных дисциплин и профессионального модуля ПМ 02, на основе изучения деятельности конкретного предприятия, учреждения, организации, приобретение практического опыта;
* привитие навыков организаторской деятельности в условиях трудового коллектива, проверка профессиональной готовности к самостоятельной трудовой деятельности, будущего специалиста.

Реализация цели и задач практики должна осуществляться с учетом сферы деятельности организации или предприятия.

По окончании практики обучающийся сдаёт отчет в соответствии с содержанием тематического плана практики и по форме, установленной в техникуме ДИТИ НИЯУ МИФИ.

Итоговая аттестация проводится в форме - дифференцированного зачёта.

Продолжительность рабочего времени обучающихся при прохождении производственной практики (по профилю специальности) определяется руководителем от базы прохождения практики и составляет 108 часов независимо от возраста обучающихся и формы обучения.

Формой отчетности обучающихся по производственной практике (по профилю специальности) являются документы, содержащие результаты и отражающие качество выполнения заданий:

* дневник производственной практики;
* отчет по производственной практике.

В ходе производственной практики у обучающихся должны быть сформированы и развиты общие и профессиональные компетенций с учетом направлений деятельности:

ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.

ОК 02. Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности.

ОК 04. Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами.

ОК 09. Использовать информационные технологии в профессиональной деятельности.

ПК 2.1. Разрабатывать требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент.

ПК 2.2. Выполнять интеграцию модулей в программное обеспечение.

ПК 2.3. Выполнять отладку программного модуля с использованием специализированных программных средств.

ПК 2.4. Осуществлять разработку тестовых наборов и тестовых сценариев для программного обеспечения.

ПК 2.5. Производить инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования.

# **ПРАКТИЧЕСКАЯ ЧАСТЬ**

# **1 ЗАДАНИЕ №1**

* 1. **Описание технологических процессов разработки и сопровождения програмного обеспечения на предприятии**

Технологический процесс разработки ПО (программного обеспечения) – это последовательность действий и этапов, которые разработчики ПО должны выполнить для создания качественного, надежного и функционального программного продукта. Такой процесс включает в себя анализ требований заказчика, проектирование, программирование, тестирование, отладку, внедрение и поддержку ПО.

Весь процесс разработки ПО включает множество деталей и этапов и может занимать от нескольких месяцев до нескольких лет в зависимости от сложности и масштабов проекта. Правильно спроектированный процесс разработки ПО позволяет существенно повысить качество и надежность программного продукта.

Сопровождение ПО (программного обеспечения) — это процесс поддержки и обеспечения работоспособности и эффективности уже разработанного программного продукта.

Сопровождение ПО осуществляется после его внедрения и запуска, и может заключаться как в аутсорсинге (возложение этого процесса на стороннюю компанию), так и в организации собственной службы технической поддержки программного продукта.

Технологический процесс разработки программного продукта в ООО «СимбирСофт»:

Данная организация создаёт программные продукты на заказ:

* заказчик обращается в организацию и предоставляет техническое задание на новый продукт;
* оформляется контракт между заказчиком и предприятием;
* с техническим заданием ознакамливаются, после передают документацию подходящим по навыкам и уровню владения определённым языкам программирования  
  разработчику;
* после создания эскиза, он передаётся заказчику на одобрение;
* после создания самого программного продукты, он идёт на приёмочное тестирование;
* при удачном тестировании и одобрении заказчиком программного продукта, разработка заканчивается, все файлы и вся докумментация передаётся заказчику.

В случае, если это прописано в контракте, организация берёт на «Сопровождение» программный продукт, обязуется его обновлять и исправлять ошибки.

* 1. **Описание основных требований к содержанию документации на предприятии**

Между заказчиком и предприятием заключается контракт.

Общие требования к документации:

* ГОСТ 19.101-77 ЕСПД – Виды программ и программных документов;
* ГОСТ 19.103-77 ЕСПД – Обозначение программ и программных документов;
* ГОСТ 19.104-78 ЕСПД – Основные надписи;
* ГОСТ 19.105-78 ЕСПД – Общие требования к программным документам;
* ГОСТ 19.106-78 ЕСПД – Общие требования к программным документам, выполненным печатным способом;
* ГОСТ 19.604-78 ЕСПД – Правила внесения изменений в программные документы, выполненные печатным способом.

Заказчик создаёт техническое задание (ГОСТ 19.201-78 ЕСПД. Техническое задание. Требования к содержанию и оформлению). В процессе создания программного продукта, создаётся руководство программиста (ГОСТ 19.503-79 ЕСПД – Руководство программиста) и руководство оператора (ГОСТ-19.505-79 ЕСПД – Руководство оператора).

* 1. **Исследование процесса внедрения программных продуктов на предприятии**

Процесс внедрения программных продуктов (ПП) — это процесс установки, настройки и интеграции ПП в систему пользователя. Цель процесса внедрения программных продуктов - обеспечить гладкое и успешное внедрение ПП и обеспечить его работоспособность и эффективность в рамках заданных требований.

Процесс внедрения программных продуктов включает в себя следующие действия:

* анализ и выбор ПП, который наилучшим образом соответствует требованиям пользователя и предоставляет необходимые функции, которые помогут решить текущие бизнес-задачи;
* подготовка к установке: настройка системы и установка необходимых компонентов ПО;
* установка ПП и проведение тестирования его работоспособности;
* интеграция ПП с существующей инфраструктурой пользователя;
* обучение пользователей: обучение сотрудников работе с новым программным обеспечением;
* проведение инструктажа по использованию системы;
* техническая поддержка: разрешение возникающих проблем и вопросов, связанных с работой ПП.

Исследование процесса внедрения программных продуктов в Simbirsoft состоит из следующих этапов:

* планирование процесса внедрения. На этом этапе происходит определение целей и задач, которые требуется выполнить в рамках внедрения ПО. Определяются ресурсы, которые потребуются, и создаются графики выполнения работ;
* подготовка. В этот период создаются все необходимые документы, проводятся обучающие курсы для сотрудников, разрабатывается инфраструктура и настраивается ПО;
* внедрение. После подготовки начинается этап непосредственного внедрения программного продукта. Происходит его установка, настройка, тестирование и обучение сотрудников;
* техническая поддержка. При работе ПО могут возникать проблемы, которые требуют решения. Этот процесс включает в себя решение технических проблем и техподдержку клиентов;
* анализ. Чтобы понять, насколько эффективно прошло внедрение программного продукта, необходимо провести анализ. На этом этапе производится оценка выполненной работы и результатов внедрения. Если необходимо, вносятся корректировки;
* оптимизация. После проведения анализа выявляются проблемы, которые требуют оптимизации процесса внедрения. На основе этих данных производится оптимизация всех этапов процесса внедрения программных продуктов в Simbirsoft.
  1. **Описание видов эксплуатационных документов, используемых на предприятии**

Эксплуатационные документы — это документация, которая дает пользователям и техническому персоналу подробную информацию об использовании и обслуживании продукта в течение его жизненного цикла.

Виды эксплуатационных документов могут включать в себя:

1. Инструкции по эксплуатации (руководство пользователя) — это документ, который содержит подробное описание функций и возможностей продукта, а также инструкции по его установке, настройке и использованию. Он может предоставляться в виде бумажного и/или электронного документа.
2. Руководство администратора — это документ, который содержит подробную информацию о конфигурации, управлении и настройке системы, включая специфические требования к оборудованию и программному обеспечению.
3. Технические руководства и спецификации — это документы, которые содержат подробные технические характеристики продукта и требования к его обслуживанию и поддержке.
4. Методические материалы о качестве и технике безопасности — это документы, которые содержат рекомендации и правила по обеспечению безопасности и качества продукта.
5. Документы по сертификации — это документы, которые содержат информацию о соответствии продукта требованиям стандартов и сертификационных программ.

На предприятие ООО «СимбирСофт» используются данные документы:

* контракт (между заказчиком и предприятием), составляется для определения обязанностей между лицами;
* техническое задание, составляется для выражения планов на программу от заказчика для программистов;
* руководство программиста, составляется для программиста, для объяснения, что нужно создать и как;
* руководство оператора, составляется для пользователей программы, содержит инструкции и рекомендации.
  1. **Описание общих требований к программному документу и обозначению программ и программных документов на предприятии**

Общие требования к программному документу:

* вся документация должна быть описана в понятных и легко доступных терминах, которые будут доступны разным пользователям;
* документация должна содержать полную информацию о ПО, включая описание архитектуры, функциональности и компонентов ПО;
* документация должна быть написана в соответствии со стандартами и требованиями, принятыми в команде разработчиков;
* вся документация должна быть актуальной и легко доступной;
* вся документация должна содержать информацию о возможных ошибках, проблемах при работе, а также обучающие материалы.

Обозначение программ и программных документов на предприятии Simbirsoft:

1. Документация должна быть структурированной и легко доступной. Вся документация должна иметь унифицированные заголовки и содержать разделы, которые заполняются соответствующей информацией.
2. Документация должна быть легко доступной и должна храниться в централизованном хранилище.
3. Необходимо использовать общепринятые нотации и символы, общие для всех проектов.
4. У многих компонентов ПО должна быть сопровождающая документация, описывающая их структуру и функциональность.
   1. **Описание порядка оформления ввода ПП в эксплуатацию на предприятии**

Для оформления ввода объекта в эксплуатацию необходимо выполнить следующие шаги:

* получить техническое заключение на соответствие объекта проектной документации;
* составить акт организационно-технической готовности объекта к эксплуатации;
* обеспечить объект необходимым техническим оснащением и транспортной доступностью;
* подготовить документы для оформления прав собственности на объект (если это предусмотрено проектной документацией);
* проверить наличие согласований и разрешений от соответствующих контролирующих органов;
* оформить акт приемки объекта в эксплуатацию и передать его в соответствующие органы;
* получить разрешение на использование объекта (если это необходимо);
* уведомить прочих заинтересованных лиц о введении объекта в эксплуатацию;
* оформить все сопутствующие документы (акты, соглашения, лицензии и др.), необходимые для обеспечения безопасности и надежности объекта в эксплуатации.

Информацию о сроках и порядке оформления ввода объекта в эксплуатацию можно найти в соответствующих законодательных актах и нормативных документах.

Порядок оформления ввода ПП в эксплуатацию на предприятии Simbirsoft включает в себя следующие этапы:

1. Подготовка к вводу ПП в эксплуатацию. На этом этапе подготавливаются все необходимые документы: инструкции по эксплуатации, руководства пользователей, технические документы и т.д.
2. Установка и настройка ПП. После тестирования ПП и обнаружения отсутствия ошибок и проблем в работе, ПП должно быть установлено и настроено согласно требованиям.
3. Тестирование работоспособности. После установки и настройки ПП, происходит тестирование его работоспособности, чтобы убедиться, что оно работает корректно и эффективно.
4. Подготовка пользователей. Пользователи должны быть обучены новым функциям и возможностям ПП и его использованию в рамках бизнес-процессов.
5. Передача в эксплуатацию. После всех тестов, внедрения и обучения пользователей происходит передача ПП в эксплуатацию.
6. Техническое обслуживание. Как только ПП введено в эксплуатацию, необходимо создать программу технического обслуживания, которая включает в себя возможность быстрого и эффективного устранения ошибок и проблем, возникающих в процессе использования ПП.
7. Обслуживание клиентов. Для того, чтобы пользователи могли оценить работу ПП и организации в целом, клиентам необходимо предоставить возможность оставлять отзывы и просить помощь по вопросам, связанным с работой программного обеспечения.

# **2 ЗАДАНИЕ №2**

**2.1 Исследование основных принципов организации тестирования программных продуктов на предприятии**

Основные принципы организации тестирования программных продуктов на предприятии Simbirsoft включают в себя:

* функциональное тестирование осуществляется на основе требований заказчика, которые максимально учитывают особенности рынка и целевой аудитории;
* для обеспечения высокой эффективности использовались уникальные тестовые методики, основанные на этапах жизненного цикла программного продукта;
* команда тестирования состоит из опытных специалистов, которые прошли сертификацию в области тестирования программного обеспечения;
* тестирование выполняется на всех этапах разработки, начиная от анализа требований и заканчивая сдачей продукта в эксплуатацию;
* все выявленные ошибки документируются и участвуют в процессе управления качеством;
* тестирование выполняется на всех платформах, для которых предназначен продукт, и настройки соответствуют реальным условиям эксплуатации;
* тестирование происходит как вручную, так и в автоматическом режиме с использованием набора специальных инструментов;
* процесс тестирования является непрерывным и улучшения происходят путем анализа данных, полученных в ходе тестирования.

Тестирование программного обеспечения на предприятии Simbirsoft является важным элементом, который позволяет компании обеспечивать высокие стандарты качества в своих продуктах и удовлетворять требованиям заказчиков. Тестирование проводится в соответствии с современными методиками и определенными принципами, что позволяет принимать быстрые и надежные решения о выпуске продукции на рынок.

* 1. **Описание стадии тестирования и виды тестовых проверок, которыми чаще всего пользуются на предприятии**

Основные этапы тестирования и виды тестовых проверок, которыми часто пользуется Simbirsoft:

1. Стадия планирования тестирования. На этой стадии составляются планы тестирования, определяются стратегии, задачи и выбираются методы, используемые для тестирования.
2. Стадия подготовки тестовых данных и окружения. На этом этапе производится подготовка тестовых данных, окружения и оборудования, которые будут использоваться в процессе тестирования.
3. Функциональное тестирование. Это проверка соответствия программного продукта требованиям функциональности и реализации.
4. Нагрузочное тестирование. Нагрузочное тестирование проверяет работоспособность и скорость программного продукта при больших нагрузках.
5. Тестирование безопасности. Тестирование безопасности проверяет наличие уязвимостей и обеспечивает безопасность приложения при работе в реальных условиях.
6. Пользовательское тестирование. Это проверка пользовательского интерфейса, удобства использования приложения и других аспектов, которые влияют на конечного пользователя.
7. Тестирование на совместимость и удобство использования. Тестирование на совместимость и удобство использования проверяет, работает ли приложение на разных операционных системах и платформах, а также удобно ли оно для пользователя.
8. Тестирование производительности. Это проверка скорости работы приложения и его способности обрабатывать большие объемы данных.

Компания Simbirsoft также может использовать другие методы и виды тестирования в зависимости от характеристик и целей разрабатываемого продукта. Общий подход Simbirsoft заключается в глубоком понимании задач и, соответственно, комплексном подходе к тестированию каждой важной составляющей программного продукта.

* 1. **Определение объектов тестирования, категорий используемых тестов и виды тестирования**

Объекты тестирования — это компоненты или функциональные блоки программного продукта, которые проверяются во время тестирования. Вся архитектура приложения декомпозируется на несколько объектов, которые проверяются на соответствие требованиям заказчика и ожидания пользователя. Объекты тестирования на предприятии Simbirsoft могут включать:

* функциональные блоки программного продукта;
* различные типы пользовательских интерфейсов;
* компоненты приложения, такие как базы данных, клиентские и серверные компоненты, модули экспорта и синхронизации данных, системы аналитики, обработки изображений и многое другое.

Категории используемых тестов включают следующие виды тестирования:

* функциональное тестирование - тестирование основных функций продукта;
* пользовательское тестирование - проверка удобства использования продукта конечными пользователями;
* нагрузочное тестирование - проверка производительности и работоспособности приложения при максимальных нагрузках;
* тестирование безопасности - проверка наличия уязвимостей и обеспечение безопасности приложения при работе в реальных условиях;
* совместимость и конфигурационное тестирование - проверка совместимости продукта с различными операционными системами и конфигурациями;
* тестирование производительности - проверка производительности и работоспособности приложения.

Виды тестирования на предприятии Simbirsoft могут включать множество других тестов в зависимости от конкретных характеристик продукта. Компания имеет высокую степень экспертизы в области различных видов тестирования, которые позволяют обеспечивать высокое качество программного обеспечения. Результаты тестирования передаются в отчетной форме клиентам вместе с возможными рекомендациями по доработке продукта для удовлетворения потребностей заказчика.

* 1. **Описание тестового набора данных**

У разработанного программного продукта были проведены тесты, результаты которых показаны в таблице 2.4.1.

Таблица 2.4.1 ¬ Результаты тестирований

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № Теста | Назначение теста | Значения исходных данных | Ожидаемый результат | Фактический результат | Вывод |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 1. | Запуск микросервиса основного API | IP и порт базы данных MongoDB и Redis | Успешный запуск без вывода ошибок | Запуск микросервиса без вывода ошибок | Успешно |
| 2. | Запуск микросервиса авторизации | IP и порт базы данных MongoDB и Redis | Успешный запуск без вывода ошибок | Запуск микросервиса без вывода ошибок | Успешно |
| 3. | Запуск микросервиса фронтенда | IP и порт API и сервиса авторизации | Успешный запуск без вывода ошибок | Запуск микросервиса без вывода ошибок | Успешно |
| 4. | Запуск всего приложения со всеми микросервисами | Отсутствует | Последовательный запуск всех микросервисов | Запуск микросервиса без вывода ошибок | Успешно |

Продолжение таблицы 2.4.1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 5. | Подключение микросервиса API к базам данных MongoDB и Redis | Строка соединения к MongoDB и Redis | Запуск без ошибок и вывод об успешном соединении и отсоединения к базам данных | Зупуск микросервиса с выводом логов об соединении и отсоединении к бд | Успешно |
| 6. | Подключение микросервиса авторизации к базам данных MongoDB и Redis | Строка соединения к MongoDB и Redis | Запуск без ошибок и вывод об успешном соединении и отсоединения к базам | Зупуск микросервиса с выводом логов об соединении и отсоединении к базам данных | Успешно |
| 7. | Отправка запроса регистрации на сервер | Ввод регистрационных данных в соответствующие поля, нажарие на кнопку «Зарегистрироваться» | Переход на страницу Авторизации, отсутствие сообщения об ошибке | Успешная регистрация, переход на форму авторизации | Успешно |
| 8. | Отправка запроса на авторизацию | Ввод действующих учётных данных пользователя | Переход на страницу с проектами | Переход на страницу с проектами | Успешно |
| 9. | Отправка несуществующих учётных данных пользователя | Ввод недостоверных учётных данных в поля «Логин» и «Пароль» | Вывод сообщения об ненайденом пользователе | Вывод сообщения об ошибки авторизации | Успешно |
| 10. | Добавление нового проекта | Ввод необходимых данных для создания проекта | Сообщение об успешном добавлении проекта, переход на страницу с проектами | Сообщение об успешном добавлении проекта, переход на страницу с проектами | Успешно |
| 11. | Редактирование существующего проекта | Ввод корректных обновлённых данных о проекте | Сообщение об удачной операции | Сообщение об удачной операции | Успешно |
| 12. | Редактирование чужого или несуществующего проекта | Ввод новых данных, замена ID проекта в URL | Сообщение об ошибки, проект не существует | Сообщение об ошибке | Успешно |
| 13. | Отображение аккаунта пользователя | Переход на страницу аккаунта | Вывод информации об аккаунте | Вывод информации об аккаунте | Успешно |
| 14. | Нормальное отображение сайта | Переход по доступным страницам сайта | Отсутствие горизонтальных полос, отображение страниц согл | Отсутствие горизонтальных полос, отображение страниц согл | Успешно |

Продолжение таблицы 2.4.1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 15. | Выход из учётной записи | Нахатие на ссылку «Выход» | Переход на страницу авторизации | Переход на страницу авторизации | Успешно |
| 16 | Отображение компонента Form | Разместить форму, указать заголовок формы | Отобржение формы на странице и её содержимого | Отобржение формы на странице и её содержимого | Успешно |
| 17 | Отображение компонента Textbox | Разместить компонент Textbox, неуказывая никаких параметров | Отображение поля ввода на тестовой странице | Отображение поля ввода на тестовой странице | Успешно |
| 18 | Отображение компонента Button | Разместить компонент Button, неуказывая никаких параметров | Отображение кнопки на тестовой странице | Отображение кнопки на тестовой странице | Успешно |
| 19 | Маршрутизация на стороне клиента | Перейти на страницу авторизации, авторизованным пользователем | Перенаправление на главную страницу | Перенаправление на главную страницу | Успешно |
| 20 | Маршрутизация на стороне клиента | Авторизоваться, перейти на страницу проектов | Переход на страницу с проектами | Переход на страницу с проектсами | Успешно |

# **3 ЗАДАНИЕ №3**

**3.1 Предметная область**

Предметная область ­ часть реального мира, рассматриваемая в пределах данного контекста. Под контекстом здесь может пониматься, например, область исследования или область, которая является объектом некоторой деятельности.

Предметная область не только определяет множество объектов, рассматриваемых в пределах одного рассуждения или в научной теории, но также их свойства, отношения, функциональные связи. При этом сама по себе предметная область является абстрактным понятием, это некоторый способ «объединения предметов в мысли».

Предметной областью этого программного продукта является процесс создания Баз Данных и их составляющие. В основу приложения входит макет базы данных, который в дальнейшем, с помощью конвертера преобразуется в действующие модели для проектов на C# с EntityFramework, или для начальной структуры настоящей БД.

Программное обеспечение «Constructum» предоставляет специалисту инструментарий для работы с базами данных:

* создавать базовые решения для дальнейшей работы в сервисе;
* визуально проектировать будующую базу данных;
* создание шаблонов, возможность не заниматься рутинными задачами, а выполнять поставленные задачи;
* создание таблиц проектируемой базы данных с указанием названий;
* добавление к существующим в проекте таблицам полей с указанием типа данных, ограничениях, индексов;
* экспорт проекта в запрос для создания структуры БД, в зависимости от выбора ядра;
* преобразование текущей версии проекта в проект C# с EntityFramework или запрос, создающий базу данных.

Из возможных минусов сервиса можно выделить:

* невозможность протестировать базу во время проектирования;
* невозможно запустить базу данных здесь и сейчас;
* нет возможности отслеживать изменения при проектировке;
* отсутствуют функционал миграции
* отсутствует командная работа над проектом.

Выделим базовые сущности данной предметной области, которые образуют структуру проектируемой ИС:

Объект «user» характеризуеться следующими свойствами: код пользователя, логин, пароль, имя, фамилия.

Объект «project» характеризуеться следующими свойствами: код проекта, название проекта, код владельца, описание.

Объект «entity» характеризуеться следующими свойствами: индитификатор сущности, название сущности.

Объект «fields» характеризуеться следующими свойствами: индитификатор поля, название, тип, является ли первичным ключом, являеться ли вротичным ключом.

Объект «token» характеризуеться следующими свойствами: индитификатор токена, индитификатор пользователя привязанный к токену, токен обновления токена.

После того, как были определены объект предметной области и их свойства, необходимо построить инфологическую модель (ER-Диаграмма), которая изображена на рисунке 3.1.1.

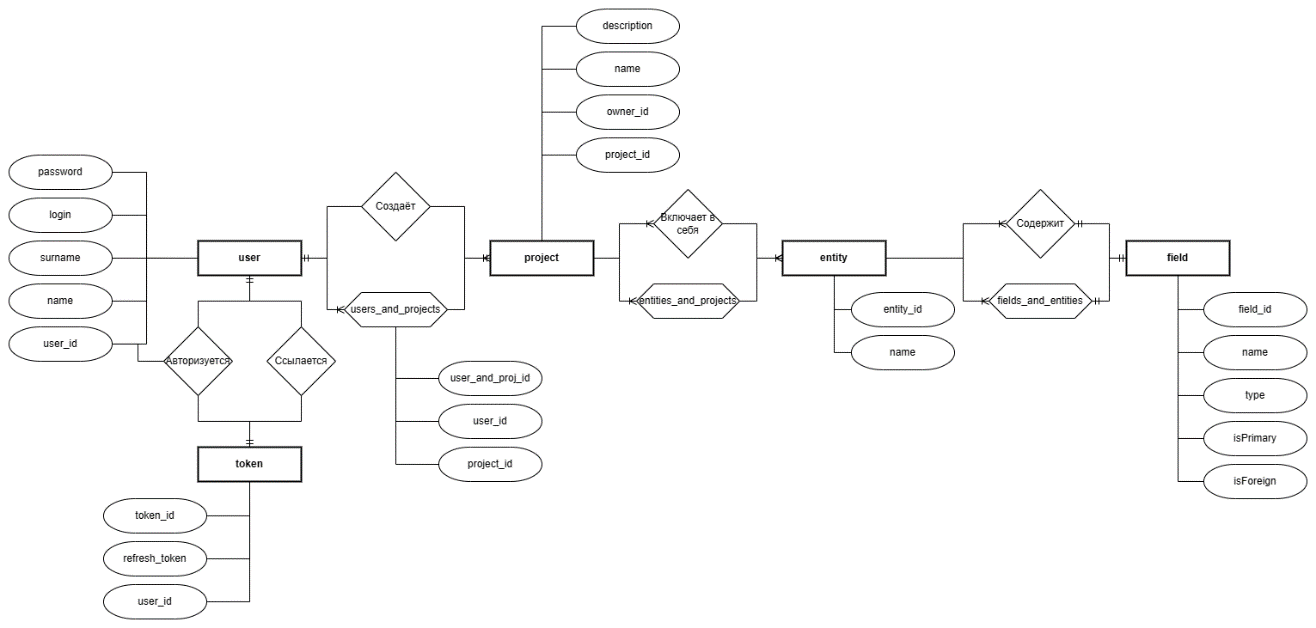


Рисунок 3.1.1 – ER-Диаграмма

На изображенной инфологической модели (ER-Диаграмма), котоорая представлена на рисунке 3.1, представлены 4 объекта и их свойства. Опишем объекты в информационной модели:

* объект «user» являеться сущностью;
* объект «token» являеться сущностью;
* объект «project» являеться сущностью;
* объект «field» являеться сущностью;
* объект «entity» являеться сущностью.

После того, как были определены объекты в инфологической модели были установлены связи между объектами информационной модели:

* сущность «user» связана с сущностью «project» один-ко-многим;
* сущность «user» связана с сущностью «token» много-комногим;
* сущность «project» связана с сущностью «entity» один-ко-многим;
* сущность «project» связана с сущностью «entity» один-ко-многим;
* сущность «entity» связана с сущностью «field» один-ко-одному;

**3.2 Описание функциональной схемы программного продукта**

Функциональная схема – это документ, разъясняющий процессы, протекающие в отдельных функциональных цепях изделия (установки) или изделия (установки) в целом.

Функциональная схема важна тем что после того как она будет построена, можно будет определить все функции программного продукта. Функциональная схема представлена на рисунке 3.2.1.

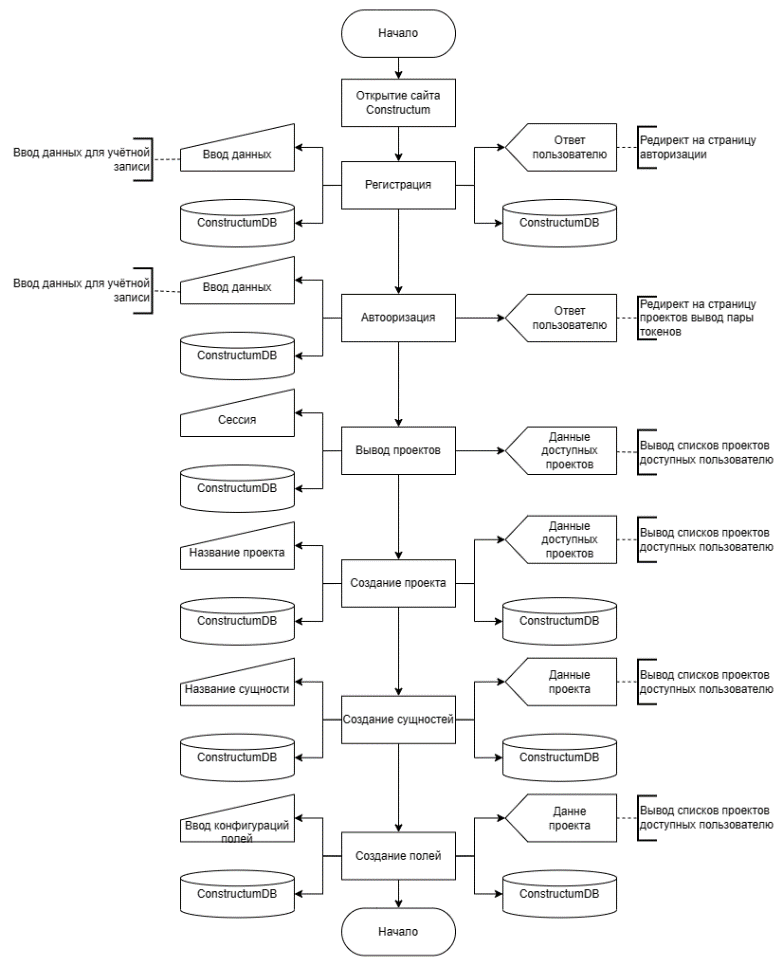


Рисунок 3.2.1 – Функциональная схема

При открытии страницы сайта «Constructum» открывается сам сайт, на котором отображаются все компоненты и представления. В ней происходит обработка и получение данных со стороны сервера.

Функция «Регистрация» – производит регистрацию пользователя в системе, использую логин, почту, пароль и подтверждение пароля.

Функция «Авторизация» – производит авторизацию пользователя в системе, использую его логин или почту и пароль.

Функция «Вывод проектов» –происходит получение и вывод данных проектов пользователю в соответствии с правами доступа установленные в проекте.

Функция «Создание проекта» –осуществляет создание проекта, совместно записывается создатель проекта.

Функция «Создание сущностей» – в соответствии с уникальным кодом проекта, появляется возможность добавить новую сущность, которая в дальнейшем будет выступать в роли таблицы для БД.

Функция «Создание полей» – отвечает за возможность добавления полей к абстрактой таблице, на этом этапе возможно указать тип и дополнительные атрибуты к полю.

**3.3 Описание модульной структуры программного продукта**

Модульная структура представляет собой иерархию процедур и функций (называемых модулями), с помощью которых программа решает поставленную задачу. Модульная структура программного продукта представлена на рисунке 3.3.1.

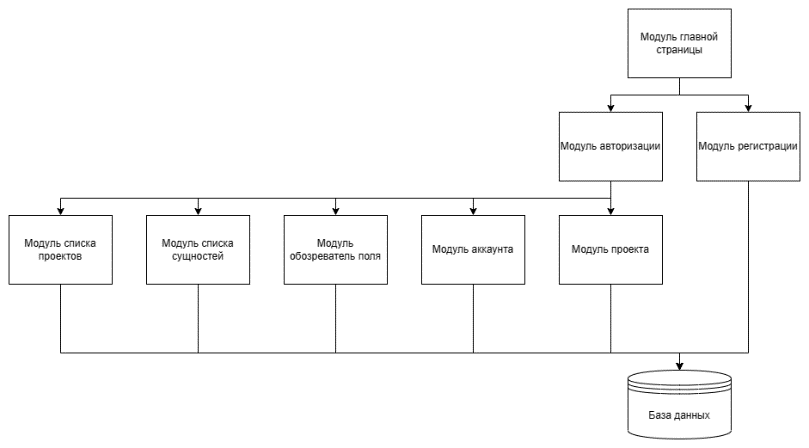


Рисунок 3.3.1 – Модульная схема

Описание модульной структуры программного продукта:

1. Модуль главной страницы – содержит название сайта, меню, ссылки на авторизацию и регистрацию.
2. Модуль авторизации – содержит два поля ввода: логин, пароль и кнопку Войти.
3. Модуль регистрации – содержит 5 поля ввода: логин, имя, фамилия, пароль, подтверждение пароля и кнопку Регистрация.
4. Модуль аккаунта – отображает загружаенные из бд информацию о пользователе.
5. Модуль списка проектов – отображает все имеющиеся проекты пользователя, а также информацию о них: название проекта, статус доступа.
6. Модуль проекта – отображает подробную информацию проекта.
7. Модуль обозреватель сущности – отображает информацию абстрактной таблицы, содержащая множество полей.
8. Модуль обозреватель поля – отображает информацию абстрактного поля, содержащее множество атрибутов.

**3.4 Описание структурной схемы программного продукта**

Структурной называют схему, отражающую состав и взаимодействие по управлению частей разрабатываемого программного обеспечения. Структурная схема определяется архитектурой разрабатываемого программного продукта. Разработку структурной схемы программы обычно выполняют методом пошаговой детализации. Компонентами структурной схемы программной системы или программного комплекса могут служить программы, подсистемы, базы данных, библиотеки ресурсов и т. п.

После того, как была разработана модульная структура, необходимо разработать структурную схему программного продукта, которая изображена на рисунке 3.4.1.

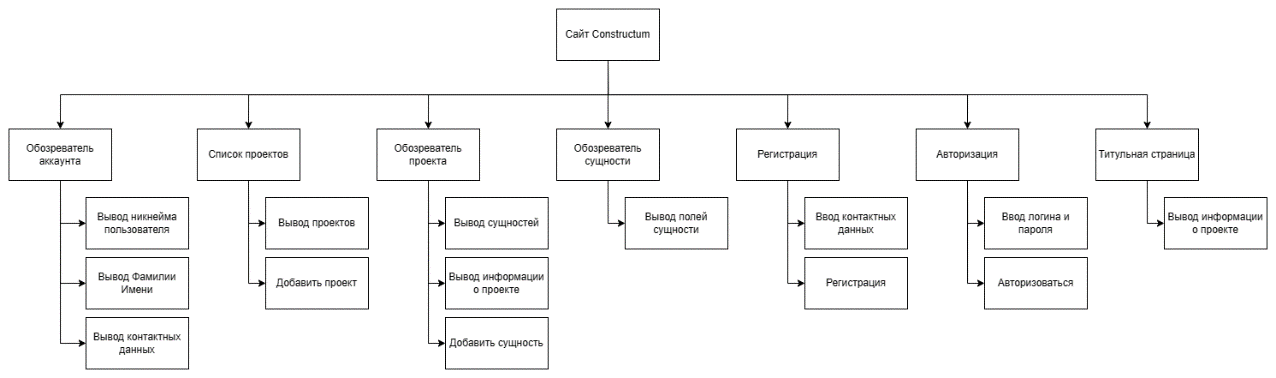


Рисунок 3.4.1 – Структурная схема

Описание структурной схемы программного продукта:

* при запуске программного продукта открывается «Титульная страница», на ней отображается информация о сайте;
* при нажатии на кнопку регистрация произойдёт переход в модуль регистрации, в которой отобразятся поля ввода и кнопка завершения регистрации;
* при нажатии на кнопку войти, произойдёт переход в модуль авторизации, в которой отображаются поля ввода логина и пароля и кнопки авторизации;
* при переходе в список проектов будут отображаться доступные пользователю проекты;
* переход в обозреватель проекта осуществляется через нажатие на кнопку подробнее;
* переход в обозреватель сущности переходит при нажатии на кнопку подробнее на странице обозревателя проекта;
* переход в обозреватель аккаунта происходит при нажатии на никнейм пользователя, в нём отображается информация пользователя.

**3.5 Описание сценария пользовательского интерфейса**

Интерфейс пользователя, он же пользовательский интерфйес – интерфейс, обеспечивающий передачу информации между пользователем – человеком и программно – аппаратными компонентами системы.

Сценарий — наглядное схематическое представление того, как пользователь решает свою задачу с помощью сайта, что ему помогает и что мешает в достижении цели. Сценарий пользовательского интерфейса представлен на рисунке 3.5.1.

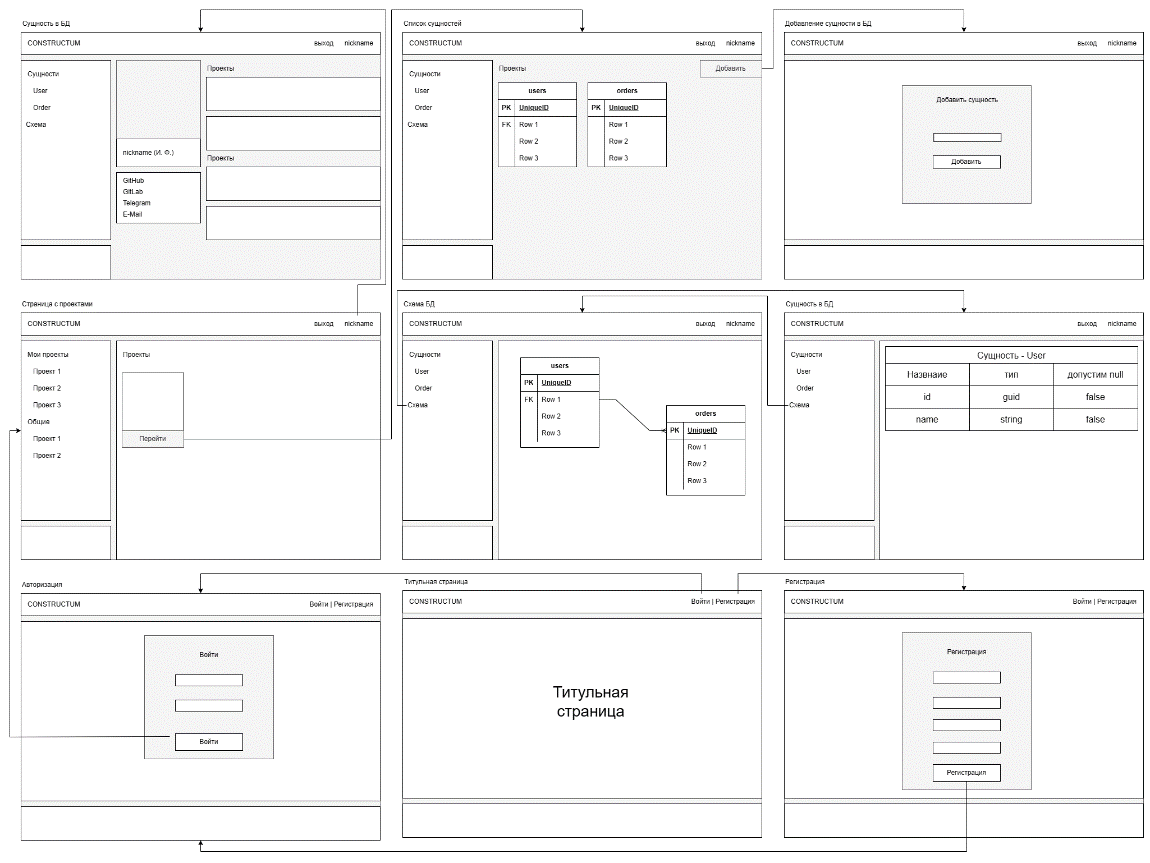


Рисунок 3.5.1 - Сценарий пользовательского интерфейса

Рассмотрим сценарий пользовательского интерфейса пользователя:

1. При запуске программного продукта открывается страница «Титульная страница», с названием компании в шапке, и пунктами «войти», «регистрация».
2. При нажатии на кнопку «войти» произойдёт переход на страницу авторизации. Которая содержит в себе поля ввода логина и пароля и кнопку Войти.
3. После успешной авторизации, произойдёт переход на главную, в шапке сайта появится никнейм.
4. При нажатии на никнейм произойдёт переход в профиль.
5. При нажатии на кнопку перейти, на странице с проектами, произойдёт переход в обозреватель проектов.
6. При нажатии на логотип произойдёт переход на главную страницу.
7. При нажатии на кнопку подробнее в карточках с проектами, произойдёт переход на страницу с выбранным проектом, где отобразятся абстрактные таблицы БД.
8. При нажатии на странице проекта кнопки «добавить сущность», произойдёт переход на страницу с добавлением сущности.
9. При нажатии в пункте меню «Схема» осуществится переход на страницу со схемой проекта. В ней будут отображены все сущности и отношения между ними.

# **4 ЗАДАНИЯ РАБОТАДАТЕЛЯ**

# **Упражнение 1**

**Постановка задачи:**

Определите переменные a, b, c и d со значениями «Привет, Simbirsoft», 2022, true, «false». При помощи оператора определения типа убедитесь, что переменные принадлежат типам: string, number, boolean. Для вывода результатов используйте функцию console.log()

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.1).

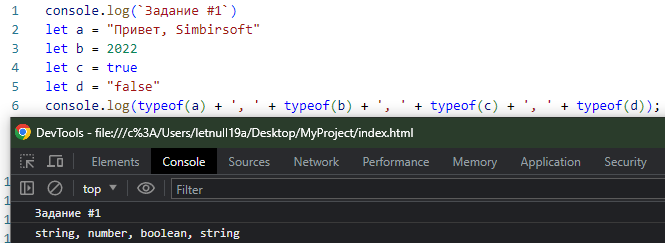


Рисунок 4.1 – Фрагмент кода и отображение

# **Упражнение 2**

**Постановка задачи:**

Создайте переменные str1='Привет, ' и str2='Мир!'. С помощью этих переменных и операции сложения строк выведите на экран фразу 'Привет, Мир!'.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.2).

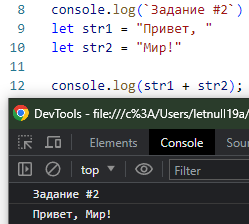


Рисунок 4.2 – Фрагмент кода и отображение

# **Упражнение 3**

**Постановка задачи:**

Создайте переменную name и присвойте ей ваше имя. Выведите на экран фразу 'Привет, %Имя%!'

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.3).

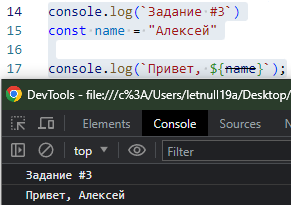


Рисунок 4.3 – Фрагмент кода и отображение

# **Упражнение 4**

**Постановка задачи:**

Создайте переменную age и присвойте ей ваш возраст. Выведите на экран 'Мне %Возраст% лет!'

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.4).

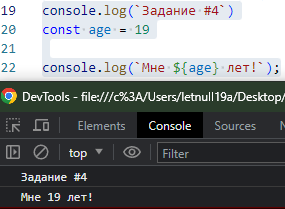


Рисунок 4.4 – Фрагмент кода и отображение

# **Упражнение 5**

**Постановка задачи:**

Создайте переменную str и присвойте ей значение 'abcde'. Обращаясь к отдельным символам этой строки выведите на экран символ 'a', символ 'c', символ 'e'.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.5).

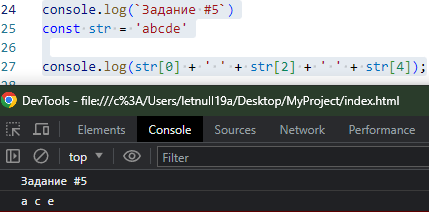


Рисунок 4.5 – Фрагмент кода и отображение

# **Упражнение 6**

**Постановка задачи:**

Создайте три переменные – час, минута, секунда. С их помощью выведите текущее время в формате 'час:минута:секунда'

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.6).

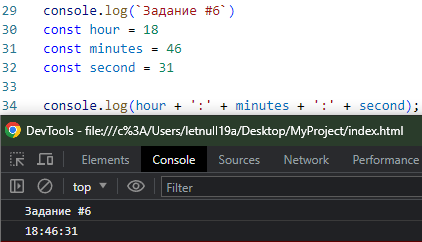


Рисунок 4.6 – Фрагмент кода и отображение

# **Упражнение 7**

**Постановка задачи:**

Создайте переменную, присвойте ей число. Извлеките квадратный корень из этого числа. Выведите его на экран, округлив до 3-х знаков после запятой.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.7).

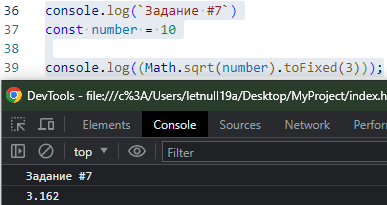


Рисунок 4.7 – Фрагмент кода и отображение

# **Упражнение 8**

**Постановка задачи:**

С сервера был получен объект user с информацией о пользователе. Выведите с помощью оператора console.log информацию о пользователе.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.8).

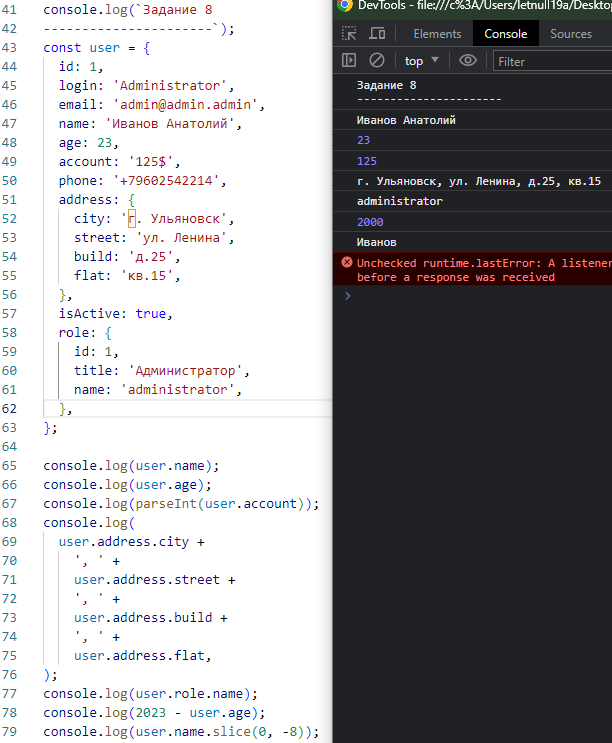


Рисунок 4.8 – Фрагмент кода и отображение

# **Упражнение 9**

**Постановка задачи:**

Если переменная a больше 3-х и меньше 10-ти, то выведите 'Верно', иначе выведите 'Неверно'.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.9).

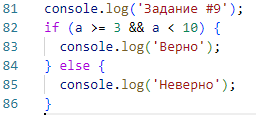


Рисунок 4.9 – Фрагмент кода

# **Упражнение 10**

**Постановка задачи:**

Если переменная a равна нулю или равна двум, то прибавьте к ней 7, иначе поделите ее на 10. Выведите новое значение переменной на экран. Проверьте работу скрипта при a, равном 5, 0, -3, 2.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.10).

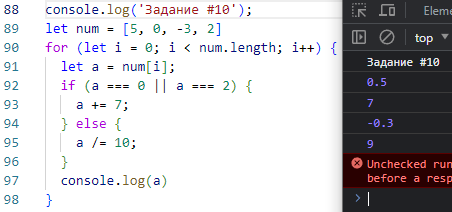


Рисунок 4.10 – Фрагмент кода и отображение

# **Упражнение 11**

**Постановка задачи:**

Переменная num может принимать 4 значения: 1, 2, 3 или 4. Если она имеет значение '1', то в переменную result запишем 'зима', если имеет значение '2' – 'весна' и так далее. Решите задачу через switch-case

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.11).

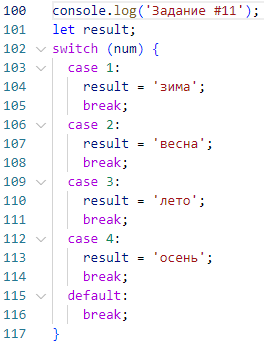


Рисунок 4.11 – Фрагмент кода

# **Упражнение 12**

**Постановка задачи:**

В переменной day лежит какое-то число из интервала от 1 до 31. Определите в какую декаду месяца попадает это число (в первую, вторую или третью)

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.12).

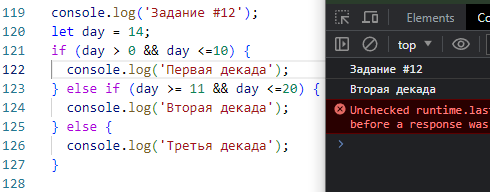


Рисунок 4.12 – Фрагмент кода и отображение

# **Упражнение 13**

**Постановка задачи:**

В переменной month лежит какое-то число из интервала от 1 до 12. Определите в какую пору года попадает этот месяц (зима, лето, весна, осень).

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.13).

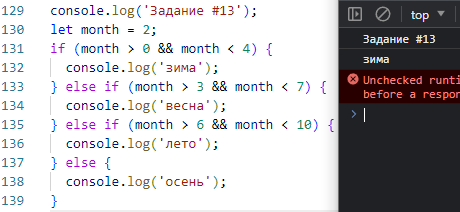


Рисунок 4.13 – Фрагмент кода и отображение

# **Упражнение 14**

**Постановка задачи:**

Дана строка, состоящая из символов, например, 'abcde'. Проверьте, что первым символом этой строки является буква 'a'. Если это так - выведите 'да', в противном случае выведите 'нет'.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.14).

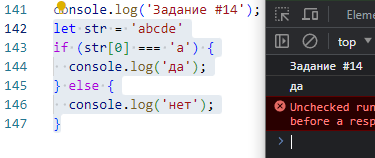


Рисунок 4.14 – Фрагмент кода и отображение

# **Упражнение 15**

**Постановка задачи:**

Есть обычная функция, которая возвращает среднее арифметическое для двух чисел. Переделайте ее в стрелочную функцию

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.15).



Рисунок 4.15 – Фрагмент кода

# **Упражнение 16**

**Постановка задачи:**

Создайте стрелочную функцию без параметров, которая выводит сообщение «Привет, мир!» в консоль.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.16).



Рисунок 4.16 – Фрагмент кода

# **Упражнение 17**

**Постановка задачи:**

Переделайте функцию из предыдущего упражнения, чтобы она принимала в качестве параметра имя и выводила в консоль «Привет, <Введенное имя>!».

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.17).



Рисунок 4.17 – Фрагмент кода

# **Упражнение 18**

**Постановка задачи:**

Найти сумму тех элементов массива, которые одновременно имеют четные и отрицательные значения. Например, в массиве [3, -5, -2, 4, -8, 0] отрицательными четными элементами являются числа -2 и -8. Их сумма равна -10.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.18).

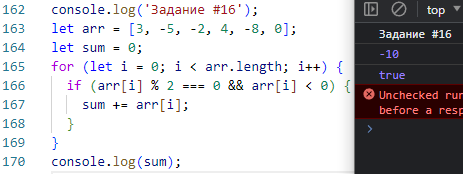


Рисунок 4.18 – Фрагмент кода и отображение

# **Упражнение 19**

**Постановка задачи:**

В массиве, состоящем из N чисел, найти максимальный по модулю элемент массива

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.19).



Рисунок 4.19 – Фрагмент кода и отображение

# **Упражнение 20**

**Постановка задачи:**

В заданном массиве чисел найти количество положительных и количество отрицательных элементов. Например, задан массив целых чисел [10, -5, 3, 2, 0, -1, 8, 0, 10, 3]. В нем 6 положительных и 2 отрицательных элемента (нули ни к тем, ни к другим не относятся)

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.20).

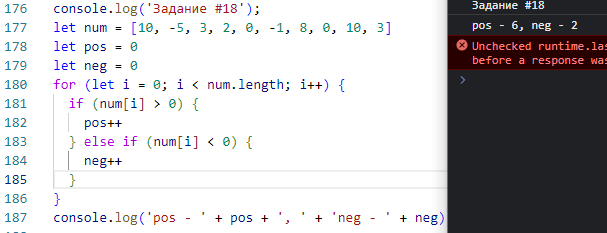


Рисунок 4.20 – Фрагмент кода и отображение

# **Упражнение 21**

**Постановка задачи:**

Дан массив целых чисел. Проверить, есть ли в нем одинаковые элементы

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.21).



Рисунок 4.21 – Фрагмент кода и отображение

# **Упражнение 22**

**Постановка задачи:**

Дан одномерный массив из 10 целых чисел. Подсчитайте количество различных чисел в нем.

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.22).



Рисунок 4.22 – Фрагмент кода и отображение

# **Упражнение 23**

**Постановка задачи:**

Напишите функцию, которая сравнивает два массива и возвращает true, если они идентичны

**Выполнение:**

Напишем JavaScript код, и запустим приложение (рис. 4.23).

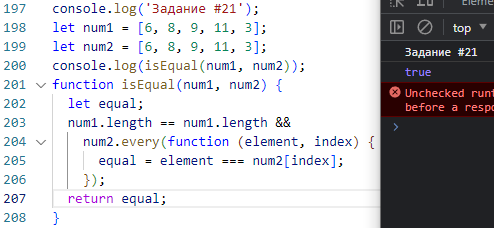


Рисунок 4.23 – Фрагмент кода и отображение

# **Упражнение 24**

**Постановка задачи:**

Создайте react-приложение с помощью Create React App используя npx и запустите сборку.

**Выполнение:**

Было создано react приложение с использованием Vite, запущенное приложение показано на (рис 4.24)

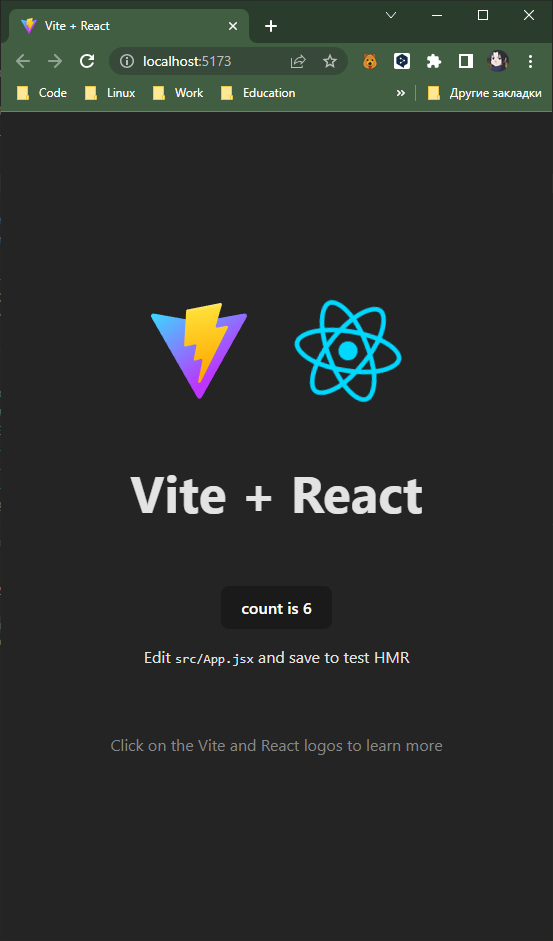


Рисунок 4.24 – Запущенное приложение

# **Упражнение 25**

**Постановка задачи:**

Создайте компонент, который будет выводить ваше имя в заголовке. Импортируйте новый компонент в App.js файл.

**Выполнение:**

Был создан компонент React который отображал вводимое в него имя (рис. 4.25).

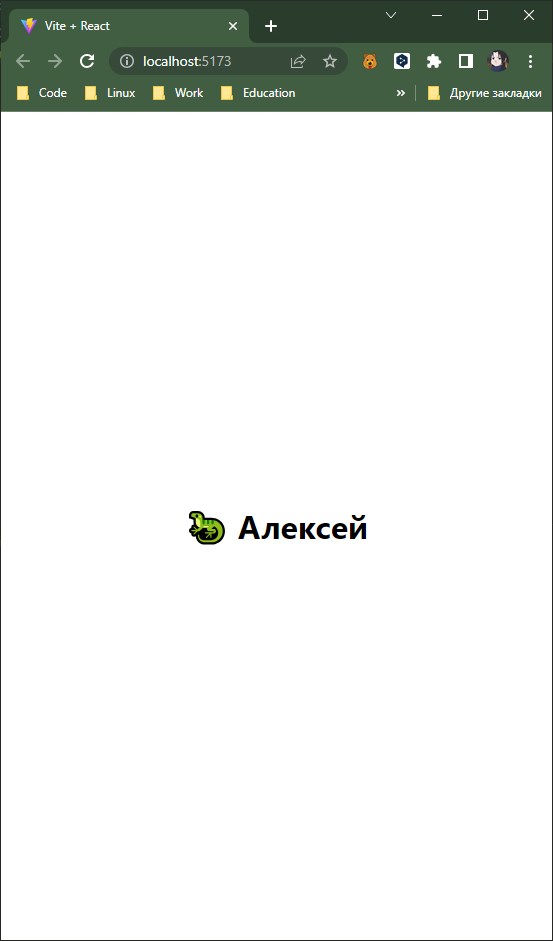


Рисунок 4.25 – Выполнение кода

# **Упражнение 26**

**Постановка задачи:**

Создайте новый компонент для отображения имя пользователя, возраст и город. '

**Выполнение:**

Был добавлен компонент который выводит данные из props (рис. 4.26).

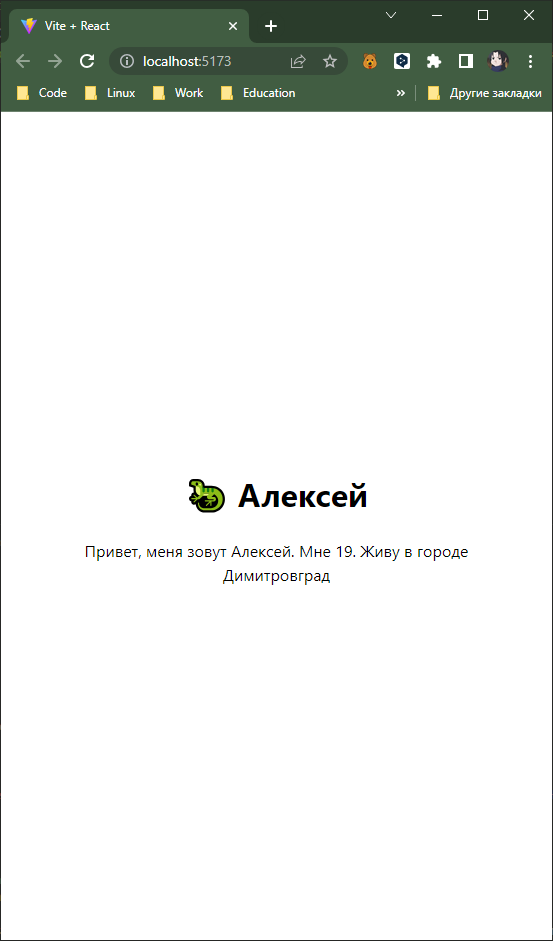


Рисунок 4.26 – Выполнение кода на странице

# **Упражнение 27**

**Постановка задачи:**

Создайте компонент, который будет выводить текст в заголовке при вводе символов в элемент Input.

**Выполнение:**

Был создан компонент представляющий из себя поле ввода, при вводе символов который отобразил бы значение в заголовке. Задача была решена двумя способами. Первый заключался в использовании состояния, второй через хук ссылки. Во втором способе не обошлось без изменения состояния. Результат представлен в (рис. 4.4).

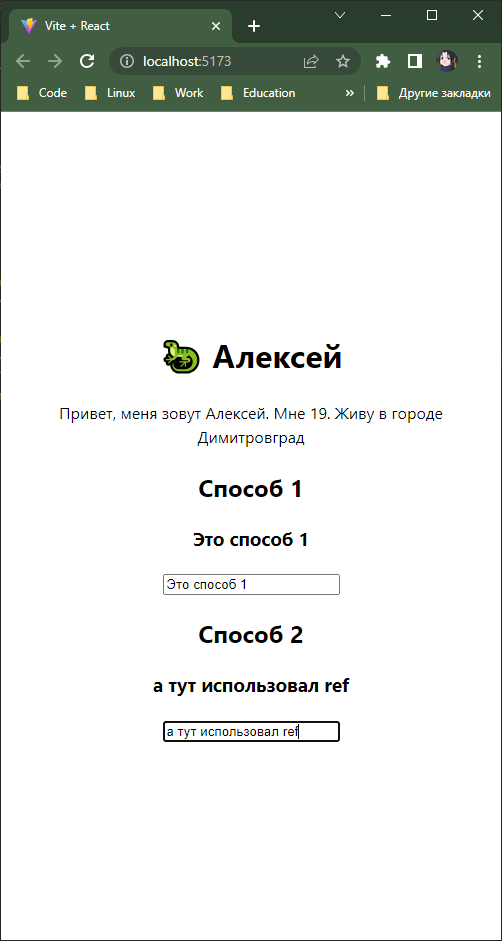


Рисунок 4.4 – Вывод данных

# **Упражнение 28**

**Постановка задачи:**

Создайте компонент, который в консоли браузера будет выводить общее количество рендеров. Добавьте два состояния text и count. Состояние count изменяется при нажатии на кнопку, а text при вводе символов в поле ввода.

**Выполнение:**

Был дополнен компонент App, в него были внесены новые состояния, в том числе и состояние перерисовки которое фиксировало сколько внесено изменений в состояниях (рис. 4.5).

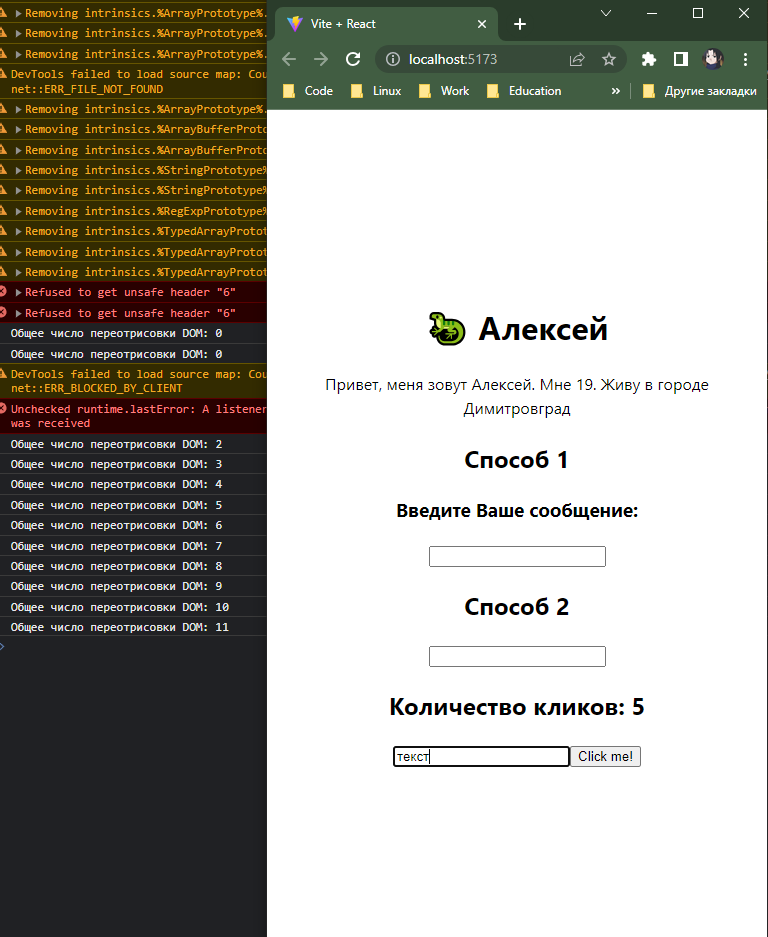


Рисунок 4.5 – Выполнение кода

# **Упражнение 29**

**Постановка задачи:**

Создайте цепочку из 5 компонентов, в родительском компоненте будет отображаться количество кликов по кнопке. В последнем компоненте создайте кнопку и через контекст передайте ей функцию для изменения состояния количество кликов.

**Выполнение:**

Были созданы дополнительно пять компонентов, каждый из которых вложен в соседний. Был объявлен контекст. В него был помещён метод для изменения состояния и само состояние. Контекст менялся в последнем компоненте при нажатии на кнопку. (рис. 4.6).

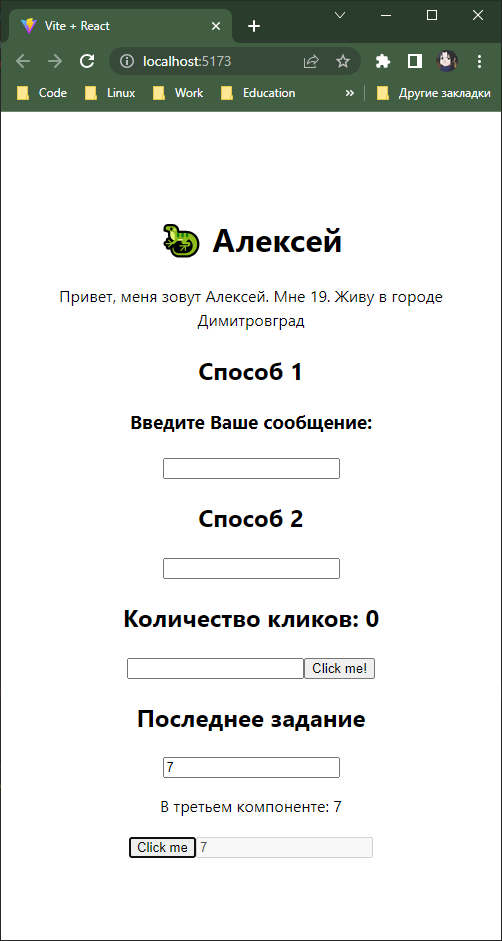


Рисунок 4.6 – Выполнение кода

# **ЗАКЛЮЧЕНИЕ**

Прохождение практики по специальности 09.02.07 «Информационные системы и программирование» проходило в организации «АИС Город», основная задача которой - оптимизация ЖКХ региона, повышение управляемости отрасли, достижение прозрачности финансовых потоков и снижения временных и моральных затрат населения.

Во время практики, были получены знания о технологических процессах разработки и сопровождения программного обеспечения, требованиях к содержанию документации, порядке проведения ввода программного продукта в эксплуатацию, а также порядке проведения, способах и методах проведения тестирования программных продуктов.

Благодаря советам и помощи руководителя, а также огромного количества, проведенного за изучением нового материала, времени, начато создание веб-сайта по предварительно разработанному макету. Получены знания о технологиях создания веб-сайтов, используемыми в настоящее время.

Помимо этого, проведено тестирование программного продукта «Фильмотека FullFilm», разработаны различные схемы и структуры на данный программный продукт, что поможет его лучшей поддержке и обновлении после введения в эксплуатацию.

В ходе прохождения производственной практики, были получены навыки:

* выбора способов решения задач профессиональной деятельности, применительно к различным контекстам;
* осуществления поиска, анализа и интерпретации информации, необходимой для выполнения задач профессиональной деятельности;
* работы в коллективе и команде, эффективного взаимодействия с коллегами, руководством, клиентами;
* использования информационных технологий в профессиональной деятельности;
* разработки требований к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонентов;
* выполнения интеграции модулей в программное обеспечение;
* выполнения отладку программного модуля с использованием специализированных программных средств;
* осуществление разработки тестовых наборов и тестовых сценариев для программного обеспечения;
* произведение инспектирования компонентов программного обеспечения на предмет соответствия стандартам кодирования.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. MDN Web Docs - сайт, содержащий расширенную документацию и примеры использования HTML, CSS и JavaScript: сайт – URL: https://developer.mozilla.org/ (дата обращения 13.06.2023).
2. Книга "Node.js в действии" - подробное руководство по Node.js, покрывающее все аспекты его использования, включая создание серверов, обработку запросов, работу с базами данных и др: сайт – URL: https://www.ozon.ru/context/detail/id/142507261/ (дата обращения 12.06.2023).
3. Учебник «JavaScript Подробное руководство» - подробное руководство по JavaScript, покрывающее все его аспекты. Включая создание интерактивных сайтов, отпарвка запросов на сервер, работу с элементами страницы и др: сайт – URL: http://kharchuk.ru/JavaScript.pdf (дата обращения 12.06.2023).
4. Официальный сайт предприятия «СимбирСофт» – контакная информация о предприятии: сайт – URL: <https://www.simbirsoft.com/> (дата обращения 07.06.2023)
5. Npm пакеты – установочные пакеты для приложений на платформе NodeJS: сайт – URL: [https://www.npmjs.com/](https://www.npmjs.com/package/npm) (дата обращения 15.06.2023)
6. Версии Javacript – инормация о совместимости версий Javacript: сайт – URL: https://msiter.ru/tutorials/javascript/js\_versions (дата обращения 10.06.2023)
7. Современный учебник JavaScript – руководство по языку программирования Javacript: сайт – URL: https://learn.javascript.ru (дата обращения 11.06.2023)
8. Руководство React – дополнительное руководство по библиотеке React: сайт – URL: https://ru.reactjs.org (дата обращения 15.06.2023)
9. Руководство React – офицальное руководство по библиотеке React: сайт – URL: https://reactdev.ru (дата обращения 15.06.2023)
10. Жизненные циклы React –руководство по жизненным циклам: сайт – URL: <https://dev-gang.ru/article/kak-ponjat-metody-zhiznennogo-cikla-komponenta-v-reactjs> (дата обращения 15.06.2023)