

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Р.Е. АЛЕКСЕЕВА»
(НГТУ)

Институт	Институт радиоэлектроники и информационных технологий
Направление подготовки	09.03.02 Информационные системы и технологии
Направленность (профиль) образовательной программы	Информационные технологии в дизайне
Кафедра	Графические информационные системы

ОТЧЕТ
по прохождению Ознакомительной практики

Выполнил Игонин
Дмитрий Евгеньевич
Студент гр. 23-ИСТ-4-1
Руководитель практики от кафедры Филинских А.Д.
Степень, звание к.т.н., доцент

Отчет защищен с оценкой
Дата защиты «19» июля 2024г.

Нижний Новгород
2024 год

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23		
Изм.	Лист	№ докум.	Подпись	Дата			
Провер.		Филинских А.Д.			Контактная форма	Лит.	Лист
Разраб.		Игонин Д.Е.					Листов
							1
							34
					ГИС-23-ИСТ-4-1		

Оглавление (Оформляется средствами WORD)

Оглавление

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ	3
Введение.	6
Разработка веб-приложения	7
Веб-приложение	7
HTML.....	10
CSS	11
JavaScript	12
Создание макета будущего веб-приложения	14
Вёрстка	16
Валидация с помощью JavaScript.....	20
Локальный сервер на Node.js	22
Заключение.....	26
Список литературы.....	27
Приложение.....	28

Разрыв этого раздела не должен переходить на другую страницу

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23		
Изм.	Лист	№ докум.	Подпись	Дата	Контактная форма		
Провер.		Филинских А.Д.					
Разраб.		Игонин Д.Е.					
					Лит.		
					Лист		
					Листов		
					2		
					34		
					ГИС-23-ИСТ-4-1		

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Р.Е. АЛЕКСЕЕВА»
(НГТУ)

Кафедра Графические информационные системы

Утверждаю:
Заведующий кафедрой
А.Д. Филинских
«06» июля 202__ г.

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

на Учебную практику (Ознакомительная практика)

Студента гр. 23-ИСТ-4-1 Ф.И.О Иголина Дмитрия Евгеньевича

Направление подготовки/специальность: 09.03.02 Информационные системы и технологии

Профиль подготовки: Информационные технологии в дизайне

Место прохождения практики НГТУ им. Р.Е. Алексеева, кафедра «Графические
информационные системы»

Время прохождения практики

Дата начала практики «06» июля 2024г.

Дата окончания практики «19» июля 2024г.

Тема индивидуального задания:

ФОРМУЛИРУЕТСЯ ТЕМА ИНДИВИДУАЛЬНОГО ЗАДАНИЯ

Содержание практики

Во время прохождения практики студент обязан:

Ознакомиться: с разделом HTML, касающийся форм и их элементов (input, textarea, button). Ознакомиться с методами CSS для стилизации формы, включая цвет, шрифты, размеры и расположение элементов. Узнать о возможностях JavaScript для добавления функциональности, такой как валидация полей и обработка события нажатия кнопки отправки.

Изучить: как проверять обязательные поля в форме, а также как валидировать формат email. Также важно разобраться в ограничении длины текстовых полей и в проверке соответствия пароля определённым критериям, таким как длина и использование различных символов. Нужно освоить обработку нажатия кнопки отправки формы с помощью слушателей событий и изучить процесс настройки и запуска сервера на платформе Node.js.

Выполнить следующие виды работ по приобретению практических навыков:

Исследовать описанные выше концепции, произвести верстку страницы на HTML, добавить необходимые стили в CSS, реализовать функциональность с помощью JavaScript и научиться практически применять Node.js.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		3

Должность на практике _____ практикант

Планируемые результаты обучения при прохождении практики

Планируемые результаты освоения образовательной программы	Планируемые результаты обучения при прохождении практики		
Код компетенции	Знать	Уметь	Владеть
ОПК-1 Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности	-основы математики, физики, вычислительной техники и программирования;	-решать стандартные профессиональные задачи с применением естественнонаучных и общеинженерных знаний, методов математического анализа и моделирования;	-навыками теоретического и экспериментального исследования объектов профессиональной деятельности;
ОПК-3 Способен решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности;	-принципы, методы и средства решения стандартных задач профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности	- решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности	-навыками подготовки обзоров, аннотаций, составления рефератов, научных докладов, публикаций и библиографии по научно-исследовательской работе с учетом требований информационной безопасности
ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий;	-основные языки программирования и работы с базами данных, операционные системы и оболочки, современные программные среды разработки информационных систем и технологий	-применять языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ	-навыками программирования, отладки и тестирования прототипов программно-технических комплексов задач

Руководитель практики от кафедры

Зав. каф. ГИС к.т.н, доцент

(ученое звание и степень)

Филинских А.Д.

(подпись)

Ф.И.О.

Задание на практику получил:

Студент

Игонин Дмитрий Евгеньевич

(подпись)

(ФИО)

«06» июля 2024 г.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист 4
Изм.	Лист	№ докум.	Подпись	Дата		

РАБОЧИЙ ГРАФИК (ПЛАН) ПРОВЕДЕНИЯ УЧЕБНОЙ ПРАКТИКИ

Студента гр. 23-ИСТ-4-1 Ф.И.О. Игонин Дмитрий Евгеньевич

№№ п/п	Разделы (этапы) практики (заполняется в соответствии с заданием)	Сроки выполнения с « <u> </u> » <u>202</u> по « <u> </u> » <u>202</u>	Отметка о выполнении (подпись руководителя практики)
1	Подготовительный (организационный) этап	06 июля 2024г 07 июля 2024г.	
1.1	Определение базы прохождения практики	06 июля 2024	
1.2	Организационное собрание для разъяснения целей, задач, содержания и порядка прохождения практики	07 июля 2024	
1.3	Получение индивидуального задания		
2	Выполнение индивидуального задания:	08 июля 2024г 17 июля 2024г	
2.1	Установка Node.js	9 июля 2024г	
2.2	Установление зависимостей Node.js	10 июля 2024г	
2.3	Разработка макета страницы (Figma)	11 июля 2024г	
2.4	Верстка страницы на html и css	12 июля 2024г	
2.5	Написание функционала с помощью JavaScript	13 июля 2024г	
2.6	Запуск сервера Node.js	15 июля 2024г	
2.6	Тестирование проекта и отладка ошибок	16 июля 2024г	
3	Заключительный этап	18 июля 2024г 19 июля 2024г	
3.1	Написание и оформление отчета по практике	18 июля 2024г	
3.2	Защита отчета по практике	19 июля 2024г	

Руководитель практики от кафедры

доцент, к.т.н. Филинских А.Д.
(ученые звание и степень) (подпись) Ф.И.О.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		5

Введение.

Веб-сайты играют важную роль в цифровом мире, обеспечивая платформу для взаимодействия между пользователями и компаниями. Одним из ключевых элементов на таких сайтах является контактная форма, которая позволяет пользователям легко и быстро отправлять свои запросы и предложения. Контактная форма — это важный инструмент, который служит связующим звеном между посетителями сайта и владельцами бизнеса. Она позволяет пользователям передавать информацию, задавать вопросы, оставлять отзывы и предлагать сотрудничество. Хорошо спроектированная контактная форма способствует улучшению пользовательского опыта, повышает доверие к сайту и стимулирует взаимодействие. В данном отчёте рассматривается процесс создания веб-сайта с контактной формой, используя современные технологии: JavaScript, Node.js и CSS.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		6

Разработка веб-приложения

Веб-приложение

Веб-приложение – это компьютерная программа, которая запускается прямо в веб-браузере. Для работы веб-приложению не требуется установка на компьютер, так что вы можете получить доступ к нему с любого устройства, имеющего веб-браузер, при условии, что оно подключено к сети Интернет.

Чем веб-приложение отличается от сайта

Веб-приложение часто путают с обычным сайтом. Однако, они имеют некоторые различия, которые мы сейчас и рассмотрим. Обычные сайты представляют собой просто набор статических страниц с информацией.

Вот примеры сайтов:

- Сайт компании Apple (<https://www.apple.com/>)
- Сайт журнала Forbes (<https://www.forbes.com/>)
- Сайт социальной сети Facebook (<https://www.facebook.com/>)

Веб-приложение в свою очередь является программой, которую можно использовать через интернет для выполнения различных задач. Оно отличается от обычного сайта тем, что позволяет пользователям взаимодействовать с ним, вводя данные и получая результаты.

Примеры веб-приложений:

- Google Maps (<https://www.google.com/maps>)
- YouTube (<https://www.youtube.com/>)
- Instagram (<https://www.instagram.com/>)

Как работает веб-приложение

Разберемся, из чего состоят веб-приложения и какие процессы происходят «под капотом».

Веб-приложения построены на основе современных технологий, таких как HTML5, CSS3, JavaScript, и фреймворках, такие как Angular, React или Vue.js.

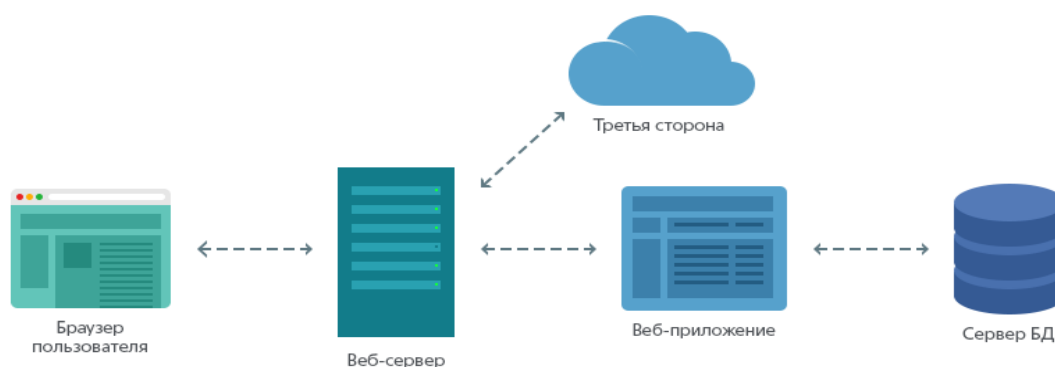
Принцип-работы веб-приложения:

Когда вы используете веб-приложение, ваш браузер отправляет запросы на сервер каждый раз, когда вы открываете страницу, нажимаете кнопку, отправляете форму, или взаимодействуете с другим элементом на странице. Сервер обрабатывает эти запросы, выполняет необходимые операции и возвращает результаты обратно.

Веб-приложения обычно используют базы данных для хранения данных. Это позволяет хранить информацию даже после того, как вы закроете свой браузер. Эти базы данных могут находиться на сервере разработчика или в

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		7

облаке



Интернет-приложения включают в себя важную составляющую в виде API (программный интерфейс веб-приложения), который позволяет им взаимодействовать со множеством сторонних систем, таких как платежные системы, CMS, социальные сети и множество других.

Виды веб-приложений

Веб-приложения бывают разных видов, и каждый из них имеет свои особенности и достоинства. В этом тексте мы поговорим о трех основных типах веб-приложений: прогрессивные веб-приложения (PWA), одностраничные приложения (SPA) и многостраничные приложения (MPA).

Прогрессивные веб-приложения (сокращенно PWA) – это новейший вид веб-приложений. Они позволяют создавать мобильное приложение из веб-версии сайта.

Самое важное преимущество прогрессивных веб-приложений заключается в том, что они могут работать даже без интернета. Это возможно благодаря Service Workers. Они кэшируют контент и обновляют его, даже если нет подключения к сети. Это значит, что пользователь может пользоваться приложением, даже если интернет отсутствует.

PWA имеют еще одно преимущество: их можно установить на устройстве пользователя. Это обеспечивает быстрый доступ к приложению без необходимости загружать его каждый раз из браузера. В результате использование PWA становится очень удобным для пользователей.

Примеры PWA:

- Google Maps
- Gmail
- Facebook Messenger

Одностраничные приложения, или SPA (на английском Single-Page Applications), — это тип веб-приложений, который загружает все необходимые ресурсы на одну страницу. Это означает, что когда пользователь переключается между разными разделами приложения, он не видит

перезагрузку страницы. Вместо этого SPA загружают новые данные и отображают их на одной и той же странице.

Главным преимуществом SPA является быстрота работы. Поскольку все данные загружаются сразу, пользователю не нужно ждать загрузки каждой отдельной страницы. Кроме того, SPA позволяет создавать более динамичные и интерактивные приложения, так как они могут обрабатывать события в режиме реального времени.

Примеры SPA:

- Airbnb
- Netflix
- Amazon

Традиционные веб-приложения состоящие из множества страниц называются МРА. Когда осуществляется переход между разделами таких приложений, происходит полное обновление страницы, что иногда замедляет работу.

Однако у традиционных веб-приложений есть и преимущества. Они отличаются большей стабильностью и надежностью, так как каждая страница при их использовании загружается отдельно и независимо от других. Это позволяет использовать различные технологии и фреймворки при разработке каждой страницы, предоставляя больше возможностей для настройки и кастомизации.

Примеры МРА:

- Wikipedia
- Ebay
- Yahoo Mail

HTML

HTML — это язык разметки гипертекстовых документов. Он нужен, чтобы отображать в браузере специальным образом отформатированный документ с множеством вложенных элементов: заголовками, абзацами, списками, гиперссылками, расположением изображений, видео и аудио.



HTML позволяет разработчикам создавать структуру веб-страниц, определяя, какие элементы будут отображаться и как они будут организованы. Это включает заголовки, абзацы, списки, таблицы, формы и медиафайлы. Структура, созданная с помощью HTML, позволяет браузерам и поисковым системам понимать и обрабатывать содержание страницы.

CSS

CSS (Cascading Style Sheets, каскадные таблицы стилей) — язык описания внешнего вида HTML-документа. Это одна из базовых технологий в современном интернете. Практически ни один сайт не обходится без CSS, поэтому HTML и CSS действуют в единой связке.

CSS



Основная идея CSS заключается в разделении содержимого и представления. Это означает, что структурная разметка содержится в HTML, а стиль и оформление — в CSS. Такое разделение позволяет упростить поддержку и обновление веб-сайтов, поскольку изменения в дизайне можно сделать, изменив только CSS, без необходимости редактировать HTML-код. CSS использует селекторы для указания на элементы HTML, к которым должны применяться стили. Эти селекторы могут быть простыми, такими как теги или классы, или сложными, комбинируя несколько условий. Например, можно применить стили к всем элементам `<p>`, к элементам с определённым классом, или к элементам, которые находятся в определённой иерархии.

Стили в CSS определяются через свойства и значения. Свойства описывают, какой аспект элемента должен быть изменён, а значения задают, каким образом это изменение должно быть выполнено. Например, свойство `color` может задавать цвет текста, а `font-size` — размер шрифта. CSS также поддерживает использование различных единиц измерения, таких как пиксели, проценты и `em`, что позволяет гибко управлять размерами и пропорциями элементов.

Одним из ключевых преимуществ CSS является возможность создания адаптивного дизайна. Адаптивный дизайн позволяет веб-страницам корректно отображаться на различных устройствах и экранах. Это достигается с помощью медиазапросов, которые позволяют применять разные стили в зависимости от характеристик устройства, таких как ширина экрана или ориентация.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		11

JavaScript

JavaScript — это высокоуровневый, динамический язык программирования, который является неотъемлемой частью современных веб-технологий наряду с HTML и CSS. В отличие от HTML, который отвечает за структуру веб-страницы, и CSS, который отвечает за её внешний вид, JavaScript обеспечивает интерактивность и динамическое поведение веб-страниц. Он позволяет создавать сложные веб-приложения с функциями, которые реагируют на действия пользователей.



JavaScript изначально был разработан как язык для браузеров, позволяющий добавлять интерактивные элементы на веб-страницы, такие как выпадающие меню, слайд-шоу и всплывающие окна. Однако со временем его возможности значительно расширились, и теперь он используется не только на клиентской стороне, но и на серверной стороне с помощью таких платформ, как Node.js.

Одной из ключевых особенностей JavaScript является его способность работать с объектной моделью документа (DOM), которая представляет собой структуру HTML-документа в виде дерева объектов. Это позволяет разработчикам динамически изменять содержимое и структуру веб-страницы без необходимости её перезагрузки. Например, можно добавлять, удалять или изменять элементы страницы в ответ на действия пользователя, такие как нажатие кнопки или ввод данных в форму.

JavaScript поддерживает асинхронное программирование, что позволяет выполнять операции, которые могут занимать длительное время (например, сетевые запросы или чтение файлов), без блокировки основной программы. Это достигается с помощью таких механизмов, как обратные вызовы (callbacks), промисы (promises) и асинхронные функции (async/await). Асинхронное программирование улучшает производительность и

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		12

отзывчивость веб-приложений, позволяя выполнять фоновые задачи параллельно с основной программой.

JavaScript также имеет богатую экосистему библиотек и фреймворков, которые упрощают разработку сложных веб-приложений. Наиболее популярные из них включают React, Angular и Vue.js, которые предоставляют инструменты и шаблоны для создания модульных и масштабируемых приложений. Эти фреймворки позволяют организовать код более структурированным и поддерживаемым образом, что особенно важно для крупных проектов.

Кроме того, благодаря Node.js, JavaScript можно использовать для серверного программирования, что позволяет создавать полные стековые приложения, где один язык используется как на клиентской, так и на серверной стороне. Node.js предоставляет высокую производительность благодаря своей асинхронной и событийно-ориентированной архитектуре, что делает его идеальным для приложений реального времени, таких как чаты и игровые серверы.

JavaScript также широко используется для создания мобильных приложений с помощью таких инструментов, как React Native, и настольных приложений с помощью Electron. Это демонстрирует универсальность и адаптивность языка, который выходит за рамки традиционного веб-разработки.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		13

Создание макета будущего веб-приложения

Figma — это мощный инструмент для дизайна интерфейсов и прототипирования, который используется профессиональными дизайнерами для создания макетов веб-страниц, мобильных приложений и других цифровых продуктов. В отличие от многих других графических редакторов, Figma работает в браузере и поддерживает совместную работу в реальном времени, что делает её идеальным выбором для командных проектов и удалённой работы.



Figma предоставляет множество инструментов для создания дизайна, включая векторные и растровые редакторы, текстовые инструменты, возможность использования слоёв и компонентов. Кроме того, она поддерживает создание интерактивных прототипов, которые позволяют визуализировать и тестировать пользовательский интерфейс до его реализации в коде.

Создание макета контактной формы в Figma включает несколько ключевых этапов. Вначале нужно определить основные элементы формы, такие как поля для ввода имени, электронной почты, сообщения и кнопки отправки. Затем можно приступить к их визуальному оформлению, используя инструменты Figma для настройки цветов, шрифтов, границ и других стилевых характеристик.

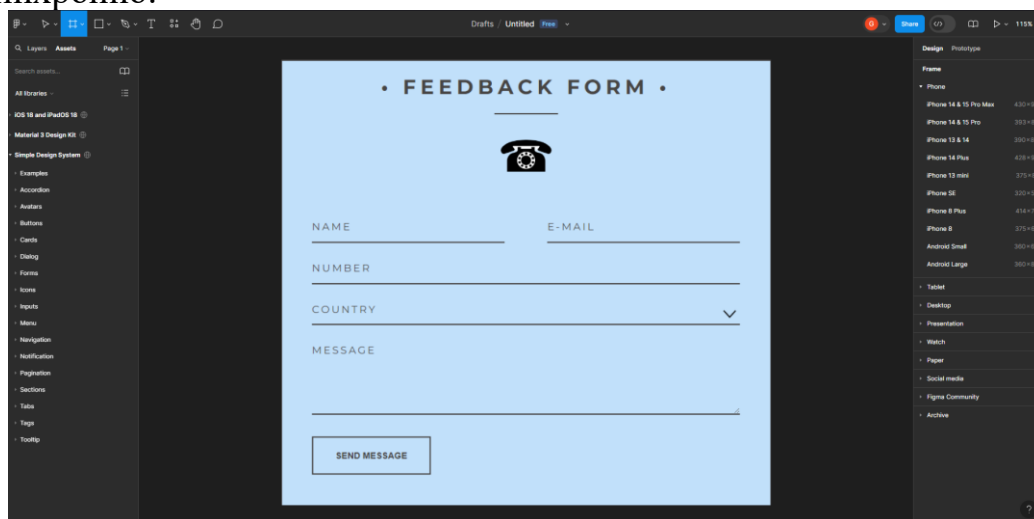
Одним из преимуществ использования Figma для создания макета является возможность работы с компонентами и стилями. Компоненты позволяют создавать повторно используемые элементы, которые можно легко обновлять во всём проекте. Например, если вы создадите кнопку отправки как компонент, вы сможете изменить её стиль в одном месте, и все экземпляры этой кнопки автоматически обновятся. Стилиевые системы позволяют задавать глобальные стили для элементов, что упрощает поддержание консистентности в дизайне.

После создания визуального макета можно настроить интерактивность, добавив кликабельные области и переходы между различными состояниями

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		14

формы. Это поможет лучше понять, как форма будет работать в реальном приложении, и выявить возможные улучшения или проблемы на раннем этапе.

Кроме того, Figma поддерживает возможность комментариев и обсуждений внутри самого макета. Это позволяет дизайнерам и разработчикам эффективно сотрудничать, обсуждать идеи и предлагать изменения прямо в контексте проекта. Возможность совместной работы в реальном времени означает, что несколько человек могут работать над одним и тем же макетом одновременно, видеть изменения друг друга и вносить правки синхронно.



Когда макет контактной формы завершён, его можно легко экспортировать в различные форматы, такие как PNG, SVG или PDF, для использования в дальнейшей разработке. Также Figma предоставляет возможность генерировать CSS-код для элементов, что упрощает процесс интеграции дизайна в реальный веб-сайт.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		15

Вёрстка

Создание контактной формы, требует продуманного подхода к структуре и стилизации.

Контактная форма состоит из следующих элементов:

- Заголовок формы.



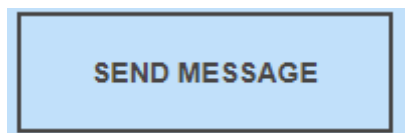
- Иконка телефона.



- Поля ввода для имени, электронной почты, номера телефона, страны и сообщения.

NAME	E-MAIL
NUMBER	
COUNTRY	▼
MESSAGE	

- Кнопка отправки.



Каждый элемент имеет свою роль и функциональность, и важно расположить их в логическом порядке внутри HTML-контейнера формы.

Для оформления формы используются стили CSS. Вот основные аспекты стилизации:

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		16

Контейнер формы: Контейнер формы имеет фон, границу и внутренние отступы, которые обеспечивают визуальное отделение формы от остального содержания страницы.

```
#container {  
  border: 3px solid #474544;  
  max-width: 768px;  
  margin: 60px auto;  
  position: relative;  
}
```

Заголовок формы: Заголовок "Feedback Form" расположен в верхней части формы и оформлен с использованием крупного шрифта, чтобы привлечь внимание пользователя.

```
Html:  
<h1>&bull; Feedback form &bull;</h1>  
Css:  
h1 {  
  color: #474544;  
  font-size: 32px;  
  font-weight: 700;  
  letter-spacing: 7px;  
  text-align: center;  
  text-transform: uppercase;  
}
```

Иконка телефона: Иконка телефона размещается под заголовком и служит визуальным элементом, указывающим на контактный характер формы.

Поля ввода: Поля ввода для имени, электронной почты, номера телефона и страны имеют одинаковую стилизацию для создания консистентного внешнего вида. Они включают тонкие границы и внутренние отступы

```
Html:  
<div class="name">  
  <label for="name"></label>  
  <input type="text" placeholder="Name" name="name" id="name_input"  
required>  
</div>  
<div class="email">  
  <label for="email"></label>  
  <input type="email" placeholder="e-mail" name="email"  
id="email_input" required>  
</div>  
<div class="telephone">  
  <label for="telephone"></label>
```

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		17

```

        <input type="text" placeholder="number" name="telephone"
id="telephone_input" required>
    </div>

```

Css:

```

input[type='text'], input[type='email'], select, textarea {
    background: none;
    border: none;
    border-bottom: 2px solid #474544;
    font-family: 'Montserrat';
    color: #474544;
    font-size: 1em;
    font-weight: 400;
    letter-spacing: 3px;
    margin: 0 0 1.875em 0;
    padding: 0 0 0.875em 0;
    text-transform: uppercase;
    width: 100%;
    box-sizing: border-box;
}
input[type='text']:focus, input[type='email']:focus, textarea:focus {
    outline: none;
}

```

Поле выбора страны: Поле выбора страны оформлено как выпадающее меню, что позволяет пользователю выбрать свою страну из списка. Стилизация этого элемента включает границы, шрифты и внутренние отступы, аналогичные полям ввода.

Html:

```

<div class="country">
    <label for="country"></label>
    <select name="country" id="country_input" required>
        <option disabled hidden selected>Country</option>
    </select>
</div>

```

Поле для сообщения: Поле для сообщения отличается от остальных полей тем, что оно больше по высоте и предназначено для ввода большого объема текста. Оно имеет аналогичную стилизацию с тонкими границами и внутренними отступами.

```

<div class="message">
    <label for="message"></label>
    <textarea name="message" placeholder="message" id="message_input"
cols="30" rows="5" required></textarea>
</div>

```

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		18

Кнопка отправки: Кнопка отправки "Send Message" расположена в нижней части формы и выделяется визуально, чтобы привлечь внимание пользователя. Она имеет стили, отличающиеся от полей ввода: более яркие цвета, рамки и отступы, а также эффект при наведении.

Html:

```
<div class="submit">  
    <input type="submit" value="Send Message" id="form_button">  
</div>
```

Css:

```
#form_button {  
    background: none;  
    border: 2px solid #474544;  
    color: #474544;  
    cursor: pointer;  
    font-size: 0.875em;  
    font-weight: bold;  
    outline: none;  
    padding: 20px 35px;  
    text-transform: uppercase;  
    transition: all 0.3s;  
}
```

```
#form_button:hover {  
    background: #474544;  
    color: #F2F3EB;  
}
```

Для написания кода использовался Visual Studio Code (VS Code) — это популярный редактор кода, разработанный Microsoft, который широко используется разработчиками для написания и редактирования кода благодаря своей мощной функциональности, поддержке различных языков программирования и интеграции с системами управления версиями, такими как Git. Выбор Visual Studio Code для разработки контактной формы предлагает множество преимуществ, которые делают процесс разработки более эффективным и приятным.



Валидация с помощью JavaScript

Валидация — это процесс проверки данных, введённых пользователем в форму, на соответствие определённым правилам или критериям. Основная цель валидации — обеспечить корректность, полноту и безопасность данных до их отправки на сервер. Валидация может выполняться как на стороне клиента (в браузере с помощью JavaScript), так и на стороне сервера.

Типы валидации

1. Клиентская валидация:

- Выполняется в браузере до отправки формы на сервер.
- Обеспечивает быстрый отклик для пользователя, показывая ошибки мгновенно.
- Реализуется с помощью HTML5 атрибутов (например, `required`, `pattern`) и JavaScript.

2. Серверная валидация:

- Выполняется на сервере после отправки данных.
- Обязательна для обеспечения безопасности, так как клиентская валидация может быть обойдена.
- Используется для проверки данных перед их обработкой или сохранением в базе данных.

Валидация контактной формы

При разработке контактной формы необходимо убедиться, что все поля заполнены корректно. Например:

- Поле номера телефона должно содержать только цифры и соответствовать определённому формату.
- Поле выбора страны должно быть обязательно заполнено и не иметь значения по умолчанию, такого как "Country".

Основные этапы кода

1. Инициализация:

- Код выполняется после загрузки HTML-документа благодаря `document.addEventListener('DOMContentLoaded', ...)`.
- Получение элементов формы и полей ввода: `form`, `telephoneInput`, `countryInput`.

2. Функции валидации:

- `validateTelephone()`: Проверяет, соответствует ли введённый номер телефона регулярному выражению (10-15 цифр с возможным ведущим плюсом).
- `validateCountry()`: Проверяет, выбрана ли страна из списка (значение не должно быть "Country").

3. Обработка отправки формы:

- При отправке формы выполняется проверка полей.
- Если одно из полей не проходит валидацию, форма не отправляется, показывается сообщение об ошибке, и соответствующее поле подсвечивается классом `.error`.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		20

4. Реакция на изменение ввода:

- Валидация номера телефона при каждом изменении ввода. Класс ошибки добавляется или убирается в зависимости от корректности ввода.
- Валидация выбора страны при изменении выбора. Класс ошибки добавляется или убирается в зависимости от выбора.

Основные функции

- **validateTelephone():**
 - Извлекает значение номера телефона.
 - Проверяет его соответствие регулярному выражению.
- **validateCountry():**
 - Проверяет, выбрано ли значение страны, отличное от "Country".
- **Обработка события submit:**
 - Проверяет оба поля.
 - Если поля некорректны, предотвращает отправку формы, показывает сообщения об ошибках и подсвечивает некорректные поля.
- **Обработка событий input и change:**
 - Валидация номера телефона при каждом изменении ввода.
 - Валидация выбора страны при изменении выбора.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
						21
Изм.	Лист	№ докум.	Подпись	Дата		

Локальный сервер на Node.js

Node.js — это среда выполнения JavaScript на серверной стороне, построенная на движке V8 от Google Chrome. Она позволяет разработчикам запускать JavaScript вне браузера, предоставляя мощные инструменты для создания масштабируемых сетевых приложений. Основной характеристикой Node.js является её однопоточная модель с асинхронной обработкой, что позволяет обрабатывать множество соединений одновременно без блокировки потока. Асинхронная природа Node.js делает её эффективной для задач ввода-вывода, таких как работа с файловой системой и сетевыми запросами.



Node.js включает в себя обширную стандартную библиотеку модулей, которые упрощают выполнение различных задач. Эти модули позволяют работать с файлами, создавать HTTP-серверы, обрабатывать потоки данных и выполнять многие другие функции. Важным компонентом Node.js является пакетный менеджер npm (Node Package Manager), который позволяет легко управлять зависимостями проекта и устанавливать необходимые пакеты из большого репозитория.

Одним из ключевых преимуществ Node.js является высокая производительность, достигаемая благодаря использованию движка V8, который компилирует JavaScript в машинный код. Масштабируемость Node.js обеспечивается его событийной моделью и асинхронной обработкой, что делает её идеальной для создания сетевых приложений. Использование JavaScript на серверной стороне также упрощает разработку, позволяя использовать единый язык для всего стека.

Node.js имеет активное сообщество разработчиков, что способствует постоянному улучшению платформы и появлению большого количества библиотек и инструментов. Она широко используется для создания веб-серверов, приложений реального времени, таких как чаты и игры, а также для микросервисной архитектуры. Node.js также часто применяется для создания серверных приложений, которые взаимодействуют с базами данных и внешними API, благодаря удобной работе с асинхронными операциями.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		22

Node.js предоставляет мощные инструменты для создания масштабируемых и эффективных серверных приложений. Её однопоточная модель, асинхронная обработка и обширная стандартная библиотека делают её идеальным выбором для различных задач, от создания веб-серверов до приложений реального времени и микросервисов.

Чтобы создать сервер нужно:

- Загрузить и установить Node.js с официального сайта, что также установит npm (Node Package Manager).
- Проверить установку, запустив команды `node -v` и `npm -v` в терминале.
- Создать новую папку для проекта.
- Перейти в эту папку через терминал и инициализировать новый проект командой `npm init -y`. Это создаст файл `package.json`, который будет содержать информацию о проекте и его зависимостях.
- Установить пакет `express`, который упрощает создание серверов в Node.js. Выполните команду `npm install express`.
- Создать файл `server.js` в корневой папке проекта. Этот файл будет содержать код вашего сервера.

Для сохранения данных формы в текстовый файл я написал следующий код:

- Для начала импортируем необходимые модули

```
const express = require('express');
const bodyParser = require('body-parser');
const fs = require('fs');
const path = require('path');
```

- Создание экземпляра приложения Express и настройка порта

```
const app = express();
const port = 3000;
```

- Настройка `body-parser` для обработки данных форм

```
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());
```

- Настройка статической папки

```
app.use(express.static(path.join(__dirname, 'public')));
```

- Обработка POST-запроса формы

```
app.post('/submit-form', (req, res) => {...}) - Обработывает POST-запросы на маршрут /submit-form.
```

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		23

`{ name, email, telephone, country, message } = req.body` - Извлекает данные формы из тела запроса.

`const formData = {...}` - Создает объект с данными формы и текущим временным штампом.

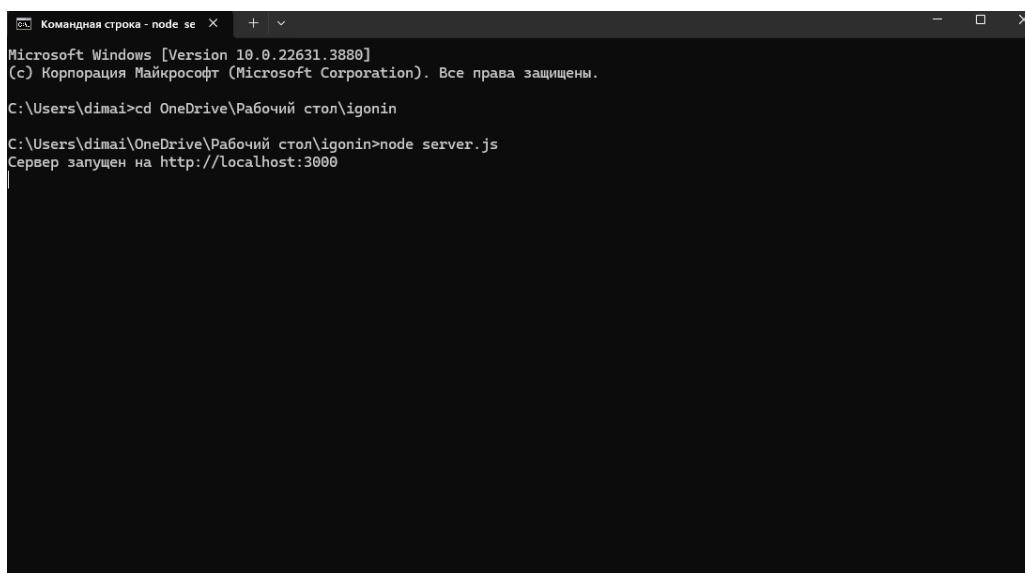
`fs.appendFile('form-data.txt', JSON.stringify(formData) + '\n', (err) => {...})` - Сохраняет данные формы в файл `form-data.txt`. Если возникает ошибка при сохранении, сервер отправляет клиенту ответ с ошибкой 500; в противном случае сервер отправляет ответ, что форма успешно отправлена.

- Запуск сервера

`app.listen(port, () => {...})`: Запускает сервер на указанном порту. Когда сервер запускается, выводится сообщение в консоль, что он работает на `http://localhost:3000`.

Чтобы запустить сервер нужно:

- Открыть командную строку и перейти в корневую папку проекта с помощью команды **cd** 'путь к проекту'
- Ввести команду **node server.js**




```
Командная строка - node se X + v
Microsoft Windows [Version 10.0.22631.3880]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\dimai>cd OneDrive\Рабочий стол\igonin

C:\Users\dimai\OneDrive\Рабочий стол\igonin>node server.js
Сервер запущен на http://localhost:3000
```

Сервер запущен. Переходим по адресу `http://localhost:3000`, вводим данные и нажимаем отправить.

• FEEDBACK FORM •



DMITRIY DIMAIGONIN@GMAIL.COM

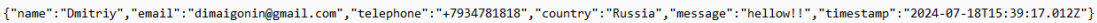
+7934781818

RUSSIA ✓

HELLOW!!

SEND MESSAGE

Заходим в form-data.txt и видим отправленные данные



```
{"name":"Dmitriy","email":"dimaigonin@gmail.com","telephone":"+7934781818","country":"Russia","message":"hellow!!","timestamp":"2024-07-18T15:39:17.012Z"}
```

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		25

Заключение

Я успешно выполнил все задачи, поставленные в начале практики по JavaScript. В процессе изучения я освоил современные методы разработки веб-приложений, фреймворки и инструменты, используемые в этой области. Это позволило мне применить практические навыки, необходимые для будущей профессиональной деятельности.

Практика по JavaScript значительно расширила мой опыт в создании веб-приложений. Я улучшил свои навыки в программировании, углубил понимание языка JavaScript и его возможностей, а также приобрел ценный опыт в процессе разработки веб-приложений.

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		26

Список литературы

Современный учебник JavaScript

<https://learn.javascript.ru/>

Node.js

<https://nodejsdev.ru/guides/freecodecamp/>

Css-tricks

<https://css-tricks.com>

Metanit

<https://metanit.com/web/javascript/>

Figma Learn

<https://help.figma.com/hc/en-us>

TutsPlus

https://code.tutsplus.com/guide-to-the-loop-in-wordpress--CRS-200905c/modifying-the-loop-using-pre_get_posts

htmlbook.ru

<http://htmlbook.ru/>

W3School

<https://www.w3schools.com/>

JavaScript.ru

<https://javascript.ru/>

HTML Academy

<https://htmlacademy.ru/>

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		27

Приложение

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Контактная форма</title>
  <link rel="stylesheet" href="css/style.css"> <!-- Подключение файла стилей -->
</head>
<body>
  <div id="container"> <!-- Контейнер для всей формы -->
    <h1>&bull; Feedback form &bull;</h1> <!-- Заголовок формы -->
    <div class="underline"></div> <!-- Линия под заголовком для стилизации -->
    <div class="icon_wrapper">
       <!-- Иконка формы -->
    </div>
    <form action="/submit-form" method="post" id="contact_form"> <!-- Начало формы,
отправка методом POST -->
      <div class="name"> <!-- Контейнер для поля имени -->
        <label for="name"></label> <!-- Метка для поля имени -->
        <input type="text" placeholder="Name" name="name" id="name_input" required> <!--
- Поле ввода имени -->
      </div>
      <div class="email"> <!-- Контейнер для поля email -->
        <label for="email"></label> <!-- Метка для поля email -->
        <input type="email" placeholder="e-mail" name="email" id="email_input" required>
<!-- Поле ввода email -->
      </div>
      <div class="telephone"> <!-- Контейнер для поля телефона -->
        <label for="telephone"></label> <!-- Метка для поля телефона -->
        <input type="text" placeholder="number" name="telephone" id="telephone_input"
required> <!-- Поле ввода телефона -->
      </div>
      <div class="country"> <!-- Контейнер для выбора страны -->
        <label for="country"></label> <!-- Метка для выбора страны -->
        <select name="country" id="country_input" required> <!-- Выпадающий список для
выбора страны -->
          <option disabled hidden selected>Country</option> <!-- Опция по умолчанию -->
          <option>Russia</option>
          <option>United States</option>
          <option>Canada</option>
          <option>United Kingdom</option>
          <option>Germany</option>
          <option>France</option>
          <option>Italy</option>
          <option>Spain</option>
          <option>China</option>
```

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		28

```

        <option>Japan</option>
        <option>Australia</option>
        <option>India</option>
        <option>Brazil</option>
        <option>Mexico</option>
        <option>South Africa</option>
        <option>Argentina</option>
        <option>Netherlands</option>
        <option>Switzerland</option>
        <option>Sweden</option>
        <option>Norway</option>
        <option>Finland</option>
        <option>Denmark</option>
        <option>Belgium</option>
        <option>Austria</option>
        <option>New Zealand</option>
        <option>South Korea</option>
        <option>Singapore</option>
    </select>
</div>
<div class="message"> <!-- Контейнер для текстового поля сообщения -->
    <label for="message"></label> <!-- Метка для текстового поля сообщения -->
    <textarea name="message" placeholder="message" id="message_input" cols="30"
rows="5" required></textarea> <!-- Текстовое поле для ввода сообщения -->
</div>
<div class="submit"> <!-- Контейнер для кнопки отправки -->
    <input type="submit" value="Send Message" id="form_button"> <!-- Кнопка
отправки формы -->
</div>
</form>
</div>

<script src="js/form.js"></script> <!-- Подключение внешнего JavaScript файла -->
</body>
</html>

```

Form.js

```

document.addEventListener('DOMContentLoaded', function() {
    // Ждём полной загрузки HTML-документа
    const form = document.getElementById('contact_form');
    // Получаем элемент формы по ID
    const telephoneInput = document.getElementById('telephone_input');
    // Получаем поле ввода телефона по ID
    const countryInput = document.getElementById('country_input');
    // Получаем поле выбора страны по ID
    const telephoneRegex = /^+?[0-9]{10,15}$/;
    // Регулярное выражение для проверки номера телефона

    function validateTelephone() {
        const telephoneValue = telephoneInput.value.trim();
    }

```

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
						29
Изм.	Лист	№ докум.	Подпись	Дата		

```

// Получаем значение телефона и убираем лишние пробелы
return telephoneRegex.test(telephoneValue);
// Проверяем значение по регулярному выражению
}

function validateCountry() {
    return countryInput.value !== "Country";
    // Проверяем, что выбрана не опция по умолчанию
}

form.addEventListener('submit', function(event) {
    // Добавляем обработчик события отправки формы
    let isValid = true;
    // Флаг для проверки валидности формы
    let errorMessage = "";
    // Строка для хранения сообщений об ошибках

    if (!validateTelephone()) {
        // Если телефон не валиден
        isValid = false;
        // Устанавливаем флаг валидности в false
        telephoneInput.classList.add('error');
        // Добавляем класс 'error' к полю телефона
        errorMessage += 'Пожалуйста, введите корректный номер телефона.\n';
        // Добавляем сообщение об ошибке
    } else {
        telephoneInput.classList.remove('error');
        // Убираем класс 'error', если поле валидно
    }

    if (!validateCountry()) {
        // Если страна не выбрана
        isValid = false;
        // Устанавливаем флаг валидности в false
        countryInput.classList.add('error');
        // Добавляем класс 'error' к полю выбора страны
        errorMessage += 'Пожалуйста, выберите страну.\n';
        // Добавляем сообщение об ошибке
    } else {
        countryInput.classList.remove('error');
        // Убираем класс 'error', если поле валидно
    }

    if (!isValid) {
        // Если форма не валидна
        event.preventDefault();
        // Предотвращаем отправку формы
        alert(errorMessage.trim());
        // Показываем сообщения об ошибках
    }
});

```

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		30

```

telephoneInput.addEventListener('input', function() {
    // Добавляем обработчик события ввода в поле телефона
    if (validateTelephone()) {
        telephoneInput.classList.remove('error');
        // Убираем класс 'error', если телефон валиден
    } else {
        telephoneInput.classList.add('error');
        // Добавляем класс 'error', если телефон не валиден
    }
});

countryInput.addEventListener('change', function() {
    // Добавляем обработчик события изменения выбора страны
    if (validateCountry()) {
        countryInput.classList.remove('error');
        // Убираем класс 'error', если страна выбрана
    } else {
        countryInput.classList.add('error');
        // Добавляем класс 'error', если страна не выбрана
    }
});
});

```

Style.css

```

@import url(https://fonts.googleapis.com/css?family=Montserrat:400,700);
/* Импорт шрифта Montserrat */

html {
    font-family: 'Montserrat', Arial;
    /* Шрифт по умолчанию */
}

body {
    background: #c1e0fa;
    /* Фоновый цвет */
}

#container {
    border: 3px solid #474544;
    max-width: 768px;
    margin: 60px auto;
    position: relative;
    /* Стил для контейнера формы */
}

form {
    padding: 37.5px;
    margin: 10px;
    /* Внутренние и внешние отступы для формы */
}

```

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
Изм.	Лист	№ докум.	Подпись	Дата		31

```

}

h1 {
  color: #474544;
  font-size: 32px;
  font-weight: 700;
  letter-spacing: 7px;
  text-align: center;
  text-transform: uppercase;
  /* Стилль для заголовка */
}

.underline {
  border-bottom: 2px solid #474544;
  margin: 0 auto;
  width: 100px;
  /* Декоративная линия под заголовком */
}

.icon_wrapper {
  margin: 40px;
  /* Отступы для обертки иконки */
}

.icon {
  display: block;
  margin: 0 auto;
  width: 80px;
  height: 50px;
  /* Стилль для иконки */
}

.email, .name {
  float: left;
  width: 45%;
  /* Стилль для полей email и name */
}

.email {
  float: right;
  /* Выравнивание email по правому краю */
}

input[type='text'], input[type='email'], select, textarea {
  background: none;
  border: none;
  border-bottom: 2px solid #474544;
  font-family: 'Montserrat';
  color: #474544;
  font-size: 1em;
  font-weight: 400;

```

					ПР-НГТУ-ИРИТ-ГИС-23-ИСТ-4-1-№05595-23	Лист
						32
Изм.	Лист	№ докум.	Подпись	Дата		


```

letter-spacing: 3px;
margin: 0 0 1.875em 0;
padding: 0 0 0.875em 0;
text-transform: uppercase;
width: 100%;
box-sizing: border-box;
/* Стиль для полей ввода, select и textarea */
}

input[type='text']:focus, input[type='email']:focus, textarea:focus {
    outline: none;
    /* Убрать обводку при фокусе */
}

select {
    background: url(select.png) no-repeat right;
    outline: none;
    -moz-appearance: none;
    -webkit-appearance: none;
    /* Стиль для выпадающего списка */
}

#form_button {
    background: none;
    border: 2px solid #474544;
    color: #474544;
    cursor: pointer;
    font-size: 0.875em;
    font-weight: bold;
    outline: none;
    padding: 20px 35px;
    text-transform: uppercase;
    transition: all 0.3s;
    /* Стиль для кнопки отправки формы */
}

#form_button:hover {
    background: #474544;
    color: #F2F3EB;
    /* Стиль для кнопки при наведении */
}

```

Server.js

```

const express = require('express');
// Импорт библиотеки Express
const bodyParser = require('body-parser');
// Импорт библиотеки body-parser для обработки данных форм
const fs = require('fs');
// Импорт модуля для работы с файловой системой
const path = require('path');
// Импорт модуля для работы с путями файловой системы

```

					Лист
					33
Изм.	Лист	№ докум.	Подпись	Дата	

```

const app = express();
// Создание экземпляра Express
const port = 3000;
// Установка порта для сервера

// Настройка body-parser для обработки данных форм
app.use(bodyParser.urlencoded({ extended: true }));
// Настройка для обработки данных форм в формате URL-encoded
app.use(bodyParser.json());
// Настройка для обработки данных форм в формате JSON

app.use(express.static(path.join(__dirname, 'public')));
// Настройка для раздачи статических файлов из директории 'public'

// Обработка POST-запроса формы
app.post('/submit-form', (req, res) => {
  const { name, email, telephone, country, message } = req.body;
  // Извлечение данных из тела запроса
  const formData = {
    name,
    email,
    telephone,
    country,
    message,
    timestamp: new Date().toISOString()
  };
  // Создание объекта с данными формы и добавление временной метки

  fs.appendFile('form-data.txt', JSON.stringify(formData) + '\n', (err) => {
    // Сохранение данных формы в файл form-data.txt
    if (err) {
      console.error('Ошибка при сохранении данных:', err);
      // Логирование ошибки в консоль
      res.status(500).send('Произошла ошибка при сохранении данных.');
```