**Курс «Создание веб-приложений с использованием**

**Angular & React»**

Тема: React: основы.

**Домашняя работа №3**

Выполнил студент гр. ВПУ-911: Мулин Николай Сергеевич

IDE: Microsoft Visual Studio Code

Ссылка на проект в GitHub: <https://github.com/Nickomu92/hw3_React>

**Условие задания**

1. *Создайте и запустите приложение React, выводящее краткую информацию*

*о вас в браузер. Например: ФИО, контактный телефон, электронный адрес.*

*При разработке нужно использовать функциональные компоненты и синтаксис JSX.*

1. *Создайте и запустите приложение React, выводящее краткую информацию*

*о вашем городе в браузер. Например: название города, название страны, год основания, несколько фотографий достопримечательностей вашего города.*

*При разработке нужно использовать функциональные компоненты и синтаксис JSX.*

1. *Создайте и запустите приложение React, выводящее информацию*

*о кулинарном рецепте в браузер. Например: название рецепта, составляющие рецепта (ингредиенты и их количество), последовательность приготовления, фотография готового блюда.*

*При разработке нужно использовать классовые компоненты и синтаксис JSX.*

1. *Используя рендеринг элементов создайте приложение, отображающее.*

*библиографию Шекспира. Создайте несколько компонентов для реализации*

*разных частей приложения. Например: компонент для отображения общей информации о Шекспире, компонент для отображения информации о конкретном произведении.*

1. *Создайте приложение «Любимый кинофильм». Оно будет содержать*

*информацию о вашем любимом фильме: название фильма, ФИО режиссера, год выпуска, киностудия, постер и т.д.*

*Обязательно используйте функциональные компоненты и props.*

1. *Создайте приложение «Персональная страница». Оно будет содержать*

*информацию о вас (ФИО, телефон, email, город проживания, опыт работы, навыки, фотографию и т.д.).*

*Обязательно используйте классовые компоненты и props.*

**Проект состоит из следующих файлов**

*(указаны только основные используемые файлы проекта, а остальные -* [*тут*](https://github.com/Nickomu92/hw3_React)*):*

***! Для стилизации использован Bootstrap5, а поскольку React-скрипты подключаем как внешние файлы (.jsx), то используем простой сервер (файл server.js), реализованный на Node.js.***

* ***index.html***

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Домашняя работа React Мулина Николая</title>

<!-- Подключаем стили Bootstrap 5 -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="styles/style.css" />

<!-- Подключаем скрипты Bootstrap 5 -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"></script>

<!-- Подключаем React -->

<script crossorigin src="https://unpkg.com/react@17/umd/react.production.min.js"></script>

<!-- Подключаем ReactDOM -->

<script crossorigin src="https://unpkg.com/react-dom@17/umd/react-dom.production.min.js"></script>

<!-- Подключаем Babel -->

<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>

</head>

<body data-bs-spy="scroll" data-bs-target=".navbar" data-bs-offset="50">

<!-- Секция для рендеринга меню (файл nav-menu.jsx) -->

<header></header>

<!-- Секция для рендеринга задания №1 (файл exercise1.jsx) -->

<section id="section1"></section>

<!-- Секция для рендеринга задания №2 (файл exercise2.jsx) -->

<section id="section2"></section>

<!-- Секция для рендеринга задания №3 (файл exercise3.jsx) -->

<section id="section3"></section>

<!-- Секция для рендеринга задания №4 (файл exercise4.jsx) -->

<section id="section4"></section>

<!-- Секция для рендеринга задания №5 (файл exercise5.jsx) -->

<section id="section5"></section>

<!-- Секция для рендеринга задания №6 (файл exercise6.jsx) -->

<section id="section6"></section>

<!-- Секция для рендеринга меню (файл footer.jsx) -->

<footer></footer>

<script src="scripts/nav-menu.jsx" type="text/babel"></script>

<script src="scripts/exercise1.jsx" type="text/babel"></script>

<script src="scripts/exercise2.jsx" type="text/babel"></script>

<script src="scripts/exercise3.jsx" type="text/babel"></script>

<script src="scripts/exercise4.jsx" type="text/babel"></script>

<script src="scripts/exercise5.jsx" type="text/babel"></script>

<script src="scripts/exercise6.jsx" type="text/babel"></script>

</body>

</html>

* ***style.css***

@import url('https://fonts.googleapis.com/css2?family=Roboto+Condensed:ital,wght@0,300;1,400&display=swap');

\* {

font-family: 'Roboto Condensed', sans-serif;

font-size: 1.2rem;

font-weight: 500;

font-style: italic;

}

body {

position: relative;

}

section {

padding: 80px 20px;

border-bottom: 2px solid grey;

}

section:nth-child(even) {

background-color: lightgray;

}

h2 {

margin: 10px auto 30px;

}

#section4 h3 {

padding: 80px 20px 20px;

}

#exercise1 {

width: 400px;

margin: 0 auto;

padding: 30px;

background-color: white;

border: 2px solid grey;

}

#section6 img {

position: relative;

width: 100%;

}

.logo {

font-size: 1.6rem;

font-weight: 900;

}

.carousel-caption {

background-color: rgba(255, 255, 255, 0.6);

color: rgb(70, 70, 70);

padding: 10px;

bottom: 10%;

}

.progress {

position: relative;

height: 30px;

}

.w-60 {

width: 60%;

}

.w-70 {

width: 70%;

}

.w-80 {

width: 80%;

}

.w-90 {

width: 90%;

}

.w-100 {

width: 100%;

}

* ***server.js***

// Простой локальный сервер на 5555 порту => localhost:5555

// Получаем модуль http, который необходим для создания сервера

const http = require("http");

// Получаем модуль fs, который необходим для работы с файлами

const fs = require("fs");

http.createServer((request, response) => {

let filePath = "index.html";

if(request.url !== "/") {

// Получаем путь после слеша

filePath = request.url.substr(1);

}

// Ищем файл

fs.readFile(filePath, function(error, data) {

// Если не нашли - ошибка

if(error) {

response.statusCode = 404;

response.end("Ресурс не найден!");

}

// Если нашли - возвращаем данные

else {

response.end(data);

}

});

// Запускаем сервер на 5555 порту

}).listen(5555, () =>

console.log("Локальный сервер запущен на localhost:5555"));

* ***nav-menu.jsx***

// Создание компонента "NavMenu" функциональный подход с ссылками-якорями для навигации по странице

function NavMenu() {

return (

<nav class="navbar navbar-expand-sm bg-dark navbar-dark fixed-top">

<div class="container-fluid">

<span class="navbar-text text-danger logo" title="Николай Мулин">Niko<span class="text-warning logo">Mu</span></span>

<ul class="navbar-nav">

<li class="nav-item">

<a class="nav-link active" href="#section1">Задание 1</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#section2">Задание 2</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#section3">Задание 3</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#section4">Задание 4</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#section5">Задание 5</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#section6">Задание 6</a>

</li>

</ul>

</div>

</nav>

)

}

// Рендеринг компонента "NavMenu"

ReactDOM.render(

<NavMenu />,

document.querySelector("header")

);

* ***exercise1.jsx***

/\*

1. Создайте и запустите приложение React, выводящее краткую информацию о вас в браузер. Например: ФИО,

контактный телефон, электронный адрес.

При разработке нужно использовать функциональные компоненты и синтаксис JSX.

\*/

// Создание компонента "Header" функциональный подход (анонимная функция)

const Header = function() {

return <h2 class="text-center text-dark">Задание 1</h2>

}

// Создание компонента "MyInfo" функциональный подход

function MyInfo() {

return (

<div id="exercise1">

<p>

<svg xmlns="http://www.w3.org/2000/svg" width="32" height="32" fill="currentColor" class="bi bi-person-circle" viewBox="0 0 20 20">

<path d="M11 6a3 3 0 1 1-6 0 3 3 0 0 1 6 0z"/>

<path fill-rule="evenodd" d="M0 8a8 8 0 1 1 16 0A8 8 0 0 1 0 8zm8-7a7 7 0 0 0-5.468 11.37C3.242 11.226 4.805 10 8 10s4.757 1.225 5.468 2.37A7 7 0 0 0 8 1z"/>

</svg>

<span className="big"> : Мулин Николай Сергеевич</span>

</p>

<p>

<svg xmlns="http://www.w3.org/2000/svg" width="32" height="32" fill="currentColor" class="bi bi-envelope-fill"viewBox="0 0 20 20">

<path d="M.05 3.555A2 2 0 0 1 2 2h12a2 2 0 0 1 1.95 1.555L8 8.414.05 3.555zM0 4.697v7.104l5.803-3.558L0 4.697zM6.761 8.83l-6.57 4.027A2 2 0 0 0 2 14h12a2 2 0 0 0 1.808-1.144l-6.57-4.027L8 9.586l-1.239-.757zm3.436-.586L16 11.801V4.697l-5.803 3.546z"/>

</svg>

<a href="https://github.com/Nickomu92"> : Nickomu92</a>

</p>

<p>

<svg xmlns="http://www.w3.org/2000/svg" width="32" height="32" fill="currentColor" class="bi bi-telephone-fill" viewBox="0 0 20 20">

<path fill-rule="evenodd" d="M1.885.511a1.745 1.745 0 0 1 2.61.163L6.29 2.98c.329.423.445.974.315 1.494l-.547 2.19a.678.678 0 0 0 .178.643l2.457 2.457a.678.678 0 0 0 .644.178l2.189-.547a1.745 1.745 0 0 1 1.494.315l2.306 1.794c.829.645.905 1.87.163 2.611l-1.034 1.034c-.74.74-1.846 1.065-2.877.702a18.634 18.634 0 0 1-7.01-4.42 18.634 18.634 0 0 1-4.42-7.009c-.362-1.03-.037-2.137.703-2.877L1.885.511z"/>

</svg>

<a href="tel:+380981111111"> : +38(098)111-11-11</a>

</p>

</div>

)

}

// Рендеринг компонентов "Header" и "MyInfo"

ReactDOM.render(

// <React.Fragment></React.Fragment> позволяет вам группировать многочисленные элементы внутри себя,

// не используя родительские элементы например div

<React.Fragment>

<Header/>

<MyInfo />

</React.Fragment>,

document.getElementById("section1")

);

* ***exercise2.jsx***

/\*

2. Создайте и запустите приложение React, выводящее краткую информацию о вашем городе в браузер.

Например: название города, название страны, год основания, несколько фотографий достопримечательностей

вашего города.

При разработке нужно использовать функциональные компоненты и синтаксис JSX.

\*/

// Создание компонента "Header" функциональный подход

function Header() {

return <h2 class="text-center text-danger">Задание 2</h2>

}

// Создание компонента "CarouselPhotos" функциональный подход (анонимная функция)

const CityInfo = function() {

return (

<>

<p>Город: <span class="text-danger">Запорожье</span>;</p>

<p>Страна: <span class="text-danger">Украина</span>;</p>

<p>Год Основания: <span class="text-danger">1770 год</span>.</p>

</>

)

}

// Создание компонента "CarouselPhotos" функциональный подход (стрелочная функция)

const CarouselPhotos = () => {

return (

<div id="demo" class="carousel slide" data-bs-ride="carousel">

<div class="carousel-indicators">

<button type="button" data-bs-target="#demo" data-bs-slide-to="0" class="active"></button>

<button type="button" data-bs-target="#demo" data-bs-slide-to="1"></button>

<button type="button" data-bs-target="#demo" data-bs-slide-to="2"></button>

<button type="button" data-bs-target="#demo" data-bs-slide-to="3"></button>

<button type="button" data-bs-target="#demo" data-bs-slide-to="4"></button>

<button type="button" data-bs-target="#demo" data-bs-slide-to="5"></button>

</div>

<div class="carousel-inner">

<div class="carousel-item active">

<img src="images/zp\_1.jpg" alt="Los Angeles" class="d-block w-100" />

<div class="carousel-caption">

<h3>Запорожье</h3>

<p>Соборный проспект вечером</p>

</div>

</div>

<div class="carousel-item">

<img src="images/zp\_2.jpg" alt="Chicago" class="d-block w-100" />

<div class="carousel-caption">

<h3>Запорожье</h3>

<p>Майдан Героев</p>

</div>

</div>

<div class="carousel-item">

<img src="images/zp\_3.jpg" alt="New York" class="d-block w-100" />

<div class="carousel-caption">

<h3>Запорожье</h3>

<p>Соборный проспект вечером</p>

</div>

</div>

<div class="carousel-item">

<img src="images/zp\_4.jpg" alt="Los Angeles" class="d-block w-100" />

<div class="carousel-caption">

<h3>Запорожье</h3>

<p>Khortitsa palace</p>

</div>

</div>

<div class="carousel-item">

<img src="images/zp\_5.jpg" alt="Chicago" class="d-block w-100" />

<div class="carousel-caption">

<h3>Запорожье</h3>

<p>Национальный университет «Запорожская политехника»</p>

</div>

</div>

<div class="carousel-item">

<img src="images/zp\_6.jpg" alt="New York" class="d-block w-100" />

<div class="carousel-caption">

<h3>Запорожье</h3>

<p>Областной украинский музыкально-драматический театр имени В.Г.Магара</p>

</div>

</div>

</div>

<button class="carousel-control-prev" type="button" data-bs-target="#demo" data-bs-slide="prev">

<span class="carousel-control-prev-icon"></span>

</button>

<button class="carousel-control-next" type="button" data-bs-target="#demo" data-bs-slide="next">

<span class="carousel-control-next-icon"></span>

</button>

</div>

)

};

// Рендеринг компонентов "Header", "CityInfo" и "CarouselPhotos"

ReactDOM.render(

<div id="container">

<Header />

<CityInfo />

<CarouselPhotos />

</div>,

document.getElementById("section2")

)

* ***exercise3.jsx***

/\*

3. Создайте и запустите приложение React, выводящее информацию о кулинарном рецепте в браузер.

Например: название рецепта, составляющие рецепта (ингредиенты и их количество), последовательность

приготовления, фотография готового блюда.

При разработке нужно использовать классовые компоненты и синтаксис JSX.

\*/

// Создание компонента "Header" классовый подход

class Header extends React.Component {

render() {

return <h2 class="text-center text-primary">Задание 3</h2>

}

}

// Создание компонента "SubHeader" классовый подход

class SubHeader extends React.Component {

render() {

return (

<React.Fragment>

<h3 class="h3 text-center">Класичний український борщ</h3>

<p>Борщ – це страва, яка є родзинкою української кухні й знана в усьому світі. Смачний, ситний, наваристий домашній борщ зігріє та об’єднає всю сім’ю за обіднім столом.</p>

</React.Fragment>

)

}

}

// Создание компонента "Ingredients" классовый подход

class Ingredients extends React.Component {

render() {

return (

<ul class="text-primary">

<li>Вода (1,5-2 л);</li>

<li>Свинина або яловичина на кістці (400 г);</li>

<li>Картопля (4 шт, середні);</li>

<li>Буряк (2 шт, невеликі);</li>

<li>Морква (1 шт);</li>

<li>Цибуля (3 шт, середні)</li>

<li>Капуста білокачанна свіжа (300 г);</li>

<li>Томатна паста (2 ст. л.);</li>

<li>Соняшникова олія (4-5 ст. л.);</li>

<li>Лимонна кислота (дрібка);</li>

<li>Сіль, лавровий лист, зелень (за смаком).</li>

</ul>

)

}

}

// Создание компонента "StepByStep" классовый подход

class StepByStep extends React.Component {

render() {

return (

<>

<p>Спершу варимо бульйон. У каструлю наливаємо 1,5-2 літра води. Додаємо м’ясо й ставимо на середній вогонь. Перед закипанням знімаємо піну. Щойно бульйон закипить, накриваємо кришкою і варимо на повільному вогні годину-півтори.</p>

<p>Тим часом готуємо засмажку. Чистимо буряк, моркву та цибулю. Буряк натираємо на крупній тертці, а моркву – на середній. Цибулю нарізаємо кубиками.</p>

<p>На середньому вогні в сковороді розігріваємо олію, висипаємо туди цибулю та моркву, смажимо 5 хвилин. Потім додаємо буряк (його можна посипати лимонною кислотою або збризнути соком свіжого лимона – так борщ буде по-справжньому червоним). Смажимо овочі ще 5 хвилин, додаємо томатну пасту, перемішуємо й смажимо все ще 5-7 хвилин.</p>

<p>А тепер варимо сам борщ. З бульйону виймаємо м’ясо і, поки воно холоне, кидаємо в бульйон нашатковану капусту. Через 5-10 хвилин додаємо нарізану соломкою картоплю. Відокремлюємо м’ясо від кістки й нарізаємо кубиками. Повертаємо м’ясо в борщ, солимо його і додаємо засмажку. Перемішуємо борщ, кладемо лавровий лист і дрібно посічену зелень, накриваємо кришкою та варимо все ще 5-7 хвилин.</p>

<p>Подаємо борщ зі сметаною і зеленню.</p>

</>

)

}

}

// Создание компонента "Photo" классовый подход

class Photo extends React.Component {

render() {

return <div class="conteiner p-3"><img src="images/borsch.jpg" alt="Борщ" /></div>

}

}

// Рендеринг компонентов "Header", "SubHeader", "Ingredients", "StepByStep" и "Photo"

ReactDOM.render(

<React.Fragment>

<Header />

<SubHeader />

<Ingredients />

<StepByStep />

<Photo />

</React.Fragment>,

document.getElementById("section3")

)

* ***exercise4.jsx***

/\*

4. Используя рендеринг элементов создайте приложение, отображающее библиографию Шекспира.

Создайте несколько компонентов для реализации разных частей приложения. Например: компонент

для отображения общей информации о Шекспире, компонент для отображения информации о конкретном произведении.

\*/

// Создание компонента "Biography" функциональный подход (анонимная функция)

const Biography = function() {

return (

<div>

<p>

Уи́льям Шекспи́р (англ. William Shakespeare, 26.04.1564 г. — 23.04.1616 г.) — английский поэт и драматург,

зачастую считается величайшим англоязычным писателем и одним из лучших драматургов мира.

Часто именуется национальным поэтом Англии.

Дошедшие до нас работы, включая некоторые, написанные совместно с другими авторами, состоят из 38 пьес,

154 сонетов, 4 поэм и 3 эпитафий. Пьесы Шекспира переведены на все основные языки и ставятся чаще, чем

произведения других драматургов.

</p>

<p>

Шекспир родился и вырос в городе Стратфорд-апон-Эйвоне. В 18 он женился на Энн Хатауэй, в браке с которой

имел трёх детей: дочь Сюзанну и двойняшек Хемнета и Джудит. Карьера Шекспира началась между 1585 и 1592

годами, когда он переехал в Лондон. Вскоре он стал успешным актёром, драматургом, а также совладельцем

театральной компании под названием «Слуги лорда-камергера», позже известной как «Слуги короля».

Около 1613 года, в возрасте 48 лет он вернулся в Стратфорд, где умер тремя годами позже. Сохранилось

мало исторических свидетельств о жизни Шекспира, и теории о его жизни создаются на основе официальных

документов и свидетельств современников, поэтому в научном сообществе до сих пор обсуждаются вопросы

относительно его внешности и религиозных воззрений, а также существует точка зрения, что приписываемые

ему работы созданы другим человеком; она популярна в культуре, хотя и отвергается подавляющим

большинством учёных-шекспироведов.

</p>

<p>

Большинство работ Шекспира написано в период с 1589 по 1613 год. Его ранние пьесы в основном относятся к

комедиям и хроникам, в которых Шекспир значительно преуспел. Затем в его творчестве настал период

трагедий, включающих произведения «Гамлет», «Король Лир», «Отелло» и «Макбет», которые считаются одними

из лучших на английском языке. В конце своего творчества Шекспир написал несколько трагикомедий, а также

сотрудничал с другими писателями.

</p>

<p>

Многие пьесы Шекспира издавались ещё при его жизни, разного качества и точности. В 1623 году два

друга Шекспира, Джон Хеминг и Генри Конделл, опубликовали Первое фолио, собрание всех, кроме двух,

пьес Шекспира, в настоящее время включаемых в канон. Том был предварен стихотворением Бена Джонсона,

в котором Джонсон провидчески приветствовал Шекспира в ныне известной цитате как «не века, но на все

времена». Позже Шекспиру различными исследователями было с различной степенью доказательности

атрибутировано ещё несколько пьес (или их фрагменты).

</p>

<p>

Уже при жизни Шекспир получал похвальные отзывы о своих работах, но по-настоящему он стал популярен

только в XIX веке. В частности, представители романтизма и викторианцы так преклонялись перед Шекспиром,

что Бернард Шоу назвал это «bardolatry» (англ.)русск., что в переводе с английского означает

«бардопоклонство». Произведения Шекспира остаются популярными и в наши дни, они постоянно изучаются

и переосмысливаются в соответствии с политическими и культурными условиями.

</p>

</div>

)

}

// Создание компонента "Menu" функциональный подход (стрелочная функция) с ссылками-якорями и навигацией по описанию произведений

const Menu = () => {

return (

<div class="container">

<nav class="navbar navbar-expand-sm bg-light navbar-success">

<span class="navbar-text text-danger">Произведения: </span>

<div class="container-fluid">

<ul class="navbar-nav">

<li class="nav-item">

<a class="nav-link active" href="#work1">"Гамлет"</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#work2">"Король Лир"</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#work3">"Отелло"</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#work4">"Макбет"</a>

</li>

</ul>

</div>

</nav>

</div>

)

}

// Создание компонента "Work1" функциональный подход (стрелочная функция)

const Work1 = () => {

return (

<div class="container">

<h3 id="work1">"Гамлет"</h3>

<p>

«Траги́ческая исто́рия о Га́млете, при́нце да́тском» (англ. The Tragical Historie of Hamlet, Prince of Denmarke)

или просто «Га́млет» — трагедия Уильяма Шекспира в пяти актах, одна из самых известных его пьес и одна из

самых знаменитых пьес в мировой драматургии. Написана в 1600—1601 годах. Это самая длинная пьеса

Шекспира — в ней 4042 строки и 29 551 слово.

</p>

<p>

Наиболее вероятная дата сочинения и первой постановки — 1600—1601 годы (театр «Глобус», Лондон).

Первый исполнитель заглавной роли — Ричард Бёрбедж; Шекспир играл тень отца Гамлета.

</p>

<p>

Трагедия входит во Всемирную библиотеку (список наиболее значимых произведений мировой литературы

Норвежского книжного клуба).

</p>

</div>

)

}

// Создание компонента "Work2" функциональный подход (стрелочная функция)

const Work2 = () => {

return (

<div class="container">

<h3 id="work2">"Король Лир"</h3>

<p>

«Коро́ль Лир» (англ. The Tragedy of King Lear) — пьеса Уильяма Шекспира, написанная в 1605—1606 годах.

Впервые напечатана в 1608 году. Основой сюжета трагедии было предание о короле Лире и его дочерях.

</p>

<p>

Пьеса была написана в ту эпоху, когда в английском обществе наблюдались перемены: старая феодальная

аристократия теряла власть и богатство, при этом пытаясь возродить прежние порядки. Возможно, что

всё это нашло своё отражение в данной трагедии: Британию безумный король разделил между герцогом

Корнуэльским и герцогом Альбанским.

</p>

</div>

)

}

// Создание компонента "Work3" функциональный подход (стрелочная функция)

const Work3 = () => {

return (

<div class="container">

<h3 id="work3">"Отелло"</h3>

<p>

«Оте́лло, венециа́нский мавр» (англ. The Tragedy of Othello, The Moor of Venice, часто просто

«Отелло» по имени главного героя) — трагедийная пьеса Уильяма Шекспира, написанная около 1604 года.

Сюжет пьесы основан на произведении Джиральди Чинтио «Венецианский мавр».

</p>

<p>

Входит во Всемирную библиотеку (список наиболее значимых произведений мировой литературы Норвежского

книжного клуба).

</p>

</div>

)

}

// Создание компонента "Work4" функциональный подход (стрелочная функция)

const Work4 = () => {

return (

<div class="container">

<h3 id="work4">"Макбет"</h3>

<p>

«Макбе́т» (англ. Macbeth) — пьеса, одна из наиболее известных трагедий Уильяма Шекспира. Пьеса,

отдалённо основанная на истории реального шотландского короля Макбета, часто представляется

архетипичной историей об опасности чрезмерной жажды власти и измены друзьям.

</p>

</div>

)

}

// Рендеринг элемента "h2" и компонентов - "Biography", "Menu", "Work1", "Work2", "Work3" и "Work4"

ReactDOM.render (

<div>

<h2 class="text-center text-warning">Задание 4</h2>

<Biography />

<Menu />

<Work1 />

<Work2 />

<Work3 />

<Work4 />

</div>,

document.getElementById("section4")

)

* ***exercise5.jsx***

/\*

5. Создайте приложение «Любимый кинофильм». Оно будет содержать информацию о вашем любимом фильме:

название фильма, ФИО режиссера, год выпуска, киностудия, постер и т.д.

Обязательно используйте функциональные компоненты и props

\*/

// Создание компонента "Header" функциональный подход (стрелочная функция)

const Header = (props) => {

return <h2 class="text-center text-success">{props.text}</h2>

}

// Создание компонента "Movie" функциональный подход (стрелочная функция)

const Movie = (props) => {

return (

<>

<h3 class="text-center">{props.title}</h3>

<p>Название: <span class="text-success">{props.name}</span>;</p>

<p>Режиссер: <span class="text-success">{props.producer}</span>;</p>

<p>Год: <span class="text-success">{props.year}</span>;</p>

<p>Киностудия: <span class="text-success">{props.filmCompany}</span></p>

<p>Постер:<br/><img src={props.image} alt="Taxi (1998) постер" class="img-thumbnail" /></p>

</>

)

}

// Рендеринг компонентов "Header" и "Movie"

ReactDOM.render(

<div className="container">

<Header text = "Задание 5"/>

<Movie title = "'Любимый кинофильм'" name="Taxi" producer="Люк Бессон" year="1998" filmCompany="Canal+" image="images/taxi\_1998\_movie.png"/>

</div>,

document.getElementById("section5")

)

* ***exercise6.jsx***

/\*

6. Создайте приложение «Персональная страница». Оно будет содержать информацию о вас (ФИО, телефон,

email, город проживания, опыт работы, навыки, фотографию и т.д.).

Обязательно используйте классовые компоненты и props.

\*/

// Создание компонента "Header" классовый подход

class Header extends React.Component {

render() {

return <h2 class="text-center text-danger">{this.props.name}</h2>

}

}

// Создание компонента "MyFullInfo" классовый подход

class MyFullInfo extends React.Component {

render() {

let lang = "progress-bar";

let phone = "tel:+";

return (

<div class="container">

<h3 class="text-center">{this.props.title}</h3>

<div class="row">

<div class="col-sm-6 mt-2">

<img src={this.props.image} alt="Аватар"/>

<p class="mt-3">

<svg xmlns="http://www.w3.org/2000/svg" width="32" height="32" fill="currentColor" class="bi bi-person-square" viewBox="0 0 20 20">

<path d="M11 6a3 3 0 1 1-6 0 3 3 0 0 1 6 0z"/>

<path d="M2 0a2 2 0 0 0-2 2v12a2 2 0 0 0 2 2h12a2 2 0 0 0 2-2V2a2 2 0 0 0-2-2H2zm12 1a1 1 0 0 1 1 1v12a1 1 0 0 1-1 1v-1c0-1-1-4-6-4s-6 3-6 4v1a1 1 0 0 1-1-1V2a1 1 0 0 1 1-1h12z"/>

</svg>

<span class="text-danger"> : {this.props.fullName}</span>;

</p>

<p>

<svg xmlns="http://www.w3.org/2000/svg" width="32" height="32" fill="currentColor" class="bi bi-telephone-fill" viewBox="0 0 20 20">

< path fill-rule="evenodd" d="M1.885.511a1.745 1.745 0 0 1 2.61.163L6.29 2.98c.329.423.445.974.315 1.494l-.547 2.19a.678.678 0 0 0 .178.643l2.457 2.457a.678.678 0 0 0 .644.178l2.189-.547a1.745 1.745 0 0 1 1.494.315l2.306 1.794c.829.645.905 1.87.163 2.611l-1.034 1.034c-.74.74-1.846 1.065-2.877.702a18.634 18.634 0 0 1-7.01-4.42 18.634 18.634 0 0 1-4.42-7.009c-.362-1.03-.037-2.137.703-2.877L1.885.511z"/>

</svg>

<a href={phone += this.props.phone}><span class="text-danger"> : +{this.props.phone}</span>;</a>

</p>

<p>

<svg xmlns="http://www.w3.org/2000/svg" width="32" height="32" fill="currentColor" class="bi bi-envelope-fill" viewBox="0 0 20 20">

<path d="M.05 3.555A2 2 0 0 1 2 2h12a2 2 0 0 1 1.95 1.555L8 8.414.05 3.555zM0 4.697v7.104l5.803-3.558L0 4.697zM6.761 8.83l-6.57 4.027A2 2 0 0 0 2 14h12a2 2 0 0 0 1.808-1.144l-6.57-4.027L8 9.586l-1.239-.757zm3.436-.586L16 11.801V4.697l-5.803 3.546z"/>

</svg>

<a href={this.props.email}><span class="text-danger"> : {this.props.email}</span>;</a>

</p>

<p>

<svg xmlns="http://www.w3.org/2000/svg" width="32" height="32" fill="currentColor" class="bi bi-building" viewBox="0 0 20 20">

<path fill-rule="evenodd" d="M14.763.075A.5.5 0 0 1 15 .5v15a.5.5 0 0 1-.5.5h-3a.5.5 0 0 1-.5-.5V14h-1v1.5a.5.5 0 0 1-.5.5h-9a.5.5 0 0 1-.5-.5V10a.5.5 0 0 1 .342-.474L6 7.64V4.5a.5.5 0 0 1 .276-.447l8-4a.5.5 0 0 1 .487.022zM6 8.694 1 10.36V15h5V8.694zM7 15h2v-1.5a.5.5 0 0 1 .5-.5h2a.5.5 0 0 1 .5.5V15h2V1.309l-7 3.5V15z"/>

<path d="M2 11h1v1H2v-1zm2 0h1v1H4v-1zm-2 2h1v1H2v-1zm2 0h1v1H4v-1zm4-4h1v1H8V9zm2 0h1v1h-1V9zm-2 2h1v1H8v-1zm2 0h1v1h-1v-1zm2-2h1v1h-1V9zm0 2h1v1h-1v-1zM8 7h1v1H8V7zm2 0h1v1h-1V7zm2 0h1v1h-1V7zM8 5h1v1H8V5zm2 0h1v1h-1V5zm2 0h1v1h-1V5zm0-2h1v1h-1V3z"/>

</svg>

<span class="text-danger"> : {this.props.city}</span>;

</p>

<p>

<svg xmlns="http://www.w3.org/2000/svg" width="32" height="32" fill="currentColor" class="bi bi-briefcase-fill" viewBox="0 0 20 20">

<path d="M6.5 1A1.5 1.5 0 0 0 5 2.5V3H1.5A1.5 1.5 0 0 0 0 4.5v1.384l7.614 2.03a1.5 1.5 0 0 0 .772 0L16 5.884V4.5A1.5 1.5 0 0 0 14.5 3H11v-.5A1.5 1.5 0 0 0 9.5 1h-3zm0 1h3a.5.5 0 0 1 .5.5V3H6v-.5a.5.5 0 0 1 .5-.5z"/>

<path d="M0 12.5A1.5 1.5 0 0 0 1.5 14h13a1.5 1.5 0 0 0 1.5-1.5V6.85L8.129 8.947a.5.5 0 0 1-.258 0L0 6.85v5.65z"/>

</svg>

<span class="text-danger"> : {this.props.workExperience}</span>.

</p>

</div>

<div class="col-sm-6">

<div class="navbar-text text-dark">Навыки: </div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.lang1Level}>{this.props.lang1}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.lang2Level}>{this.props.lang2}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.lang3Level}>{this.props.lang3}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.lang4Level}>{this.props.lang4}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.lang5Level}>{this.props.lang5}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.lang6Level}>{this.props.lang6}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.lang7Level}>{this.props.lang7}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.tech1Level + " bg-danger"}>{this.props.tech1}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.tech2Level + " bg-danger"}>{this.props.tech2}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.tech3Level + " bg-warning"}>{this.props.tech3}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.tech4Level + " bg-warning"}>{this.props.tech4}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.tech5Level + " bg-warning"}>{this.props.tech5}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.lib1Level + " bg-success"}>{this.props.lib1}</div>

</div>

<div class="progress mt-3">

<div class={lang + " w-" + this.props.lib2Level + " bg-success"}>{this.props.lib2}</div>

</div>

</div>

</div>

</div>

)

}

}

// Рендеринг компонентов "Header" и "MyFullInfo"

ReactDOM.render(

<>

<Header name = "Задание 6" />

<MyFullInfo title = '"Персональная страница"'

fullName = "Мулин Николай Сергеевич"

phone = "380981111111"

email = "https://github.com/Nickomu92"

city = "Запорожье"

workExperience = "2 года"

lang1 = "C++"

lang2 = "C#"

lang3 = "JavaScript"

lang4 = "PHP"

lang5 = "SQL"

lang6 = "HTML"

lang7 = "CSS"

tech1 = "Windows Forms"

tech2 = "WPF"

tech3 = "ASP.Net"

tech4 = "Angular"

tech5 = "Bootstrap"

lib1 = "React"

lib2 = "JQuery"

lang1Level = "80"

lang2Level = "90"

lang3Level = "80"

lang4Level = "80"

lang5Level = "70"

lang6Level = "90"

lang7Level = "80"

tech1Level = "70"

tech2Level = "80"

tech3Level = "70"

tech4Level = "70"

tech5Level = "70"

lib1Level = "80"

lib2Level = "60"

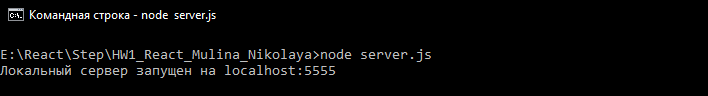
image = "images/smile.jpg"/>

</>,

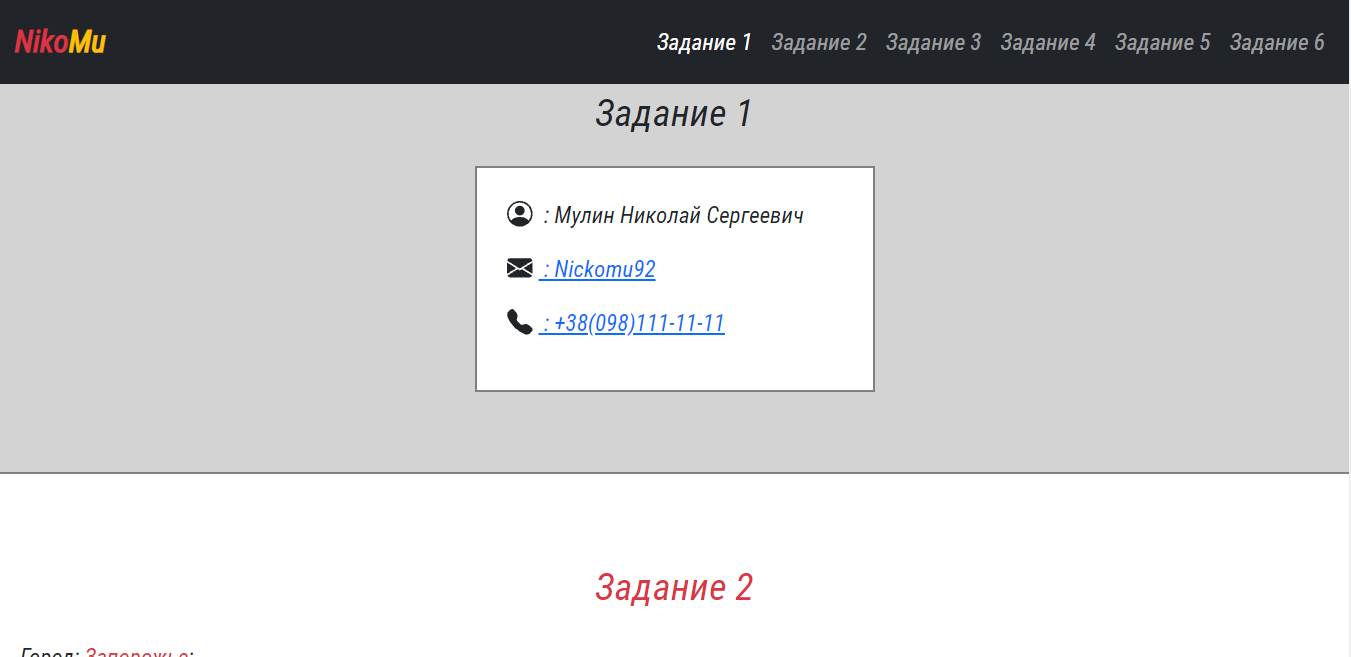
document.getElementById("section6")

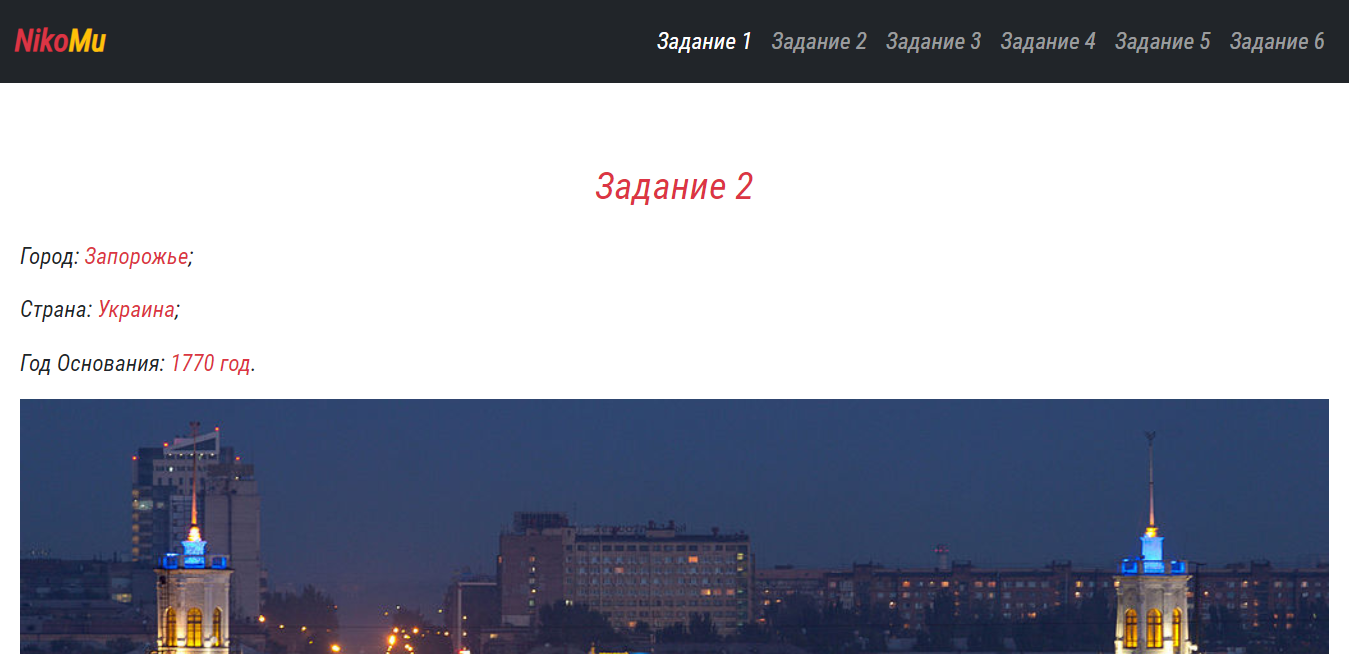
);

**Результат работы:**

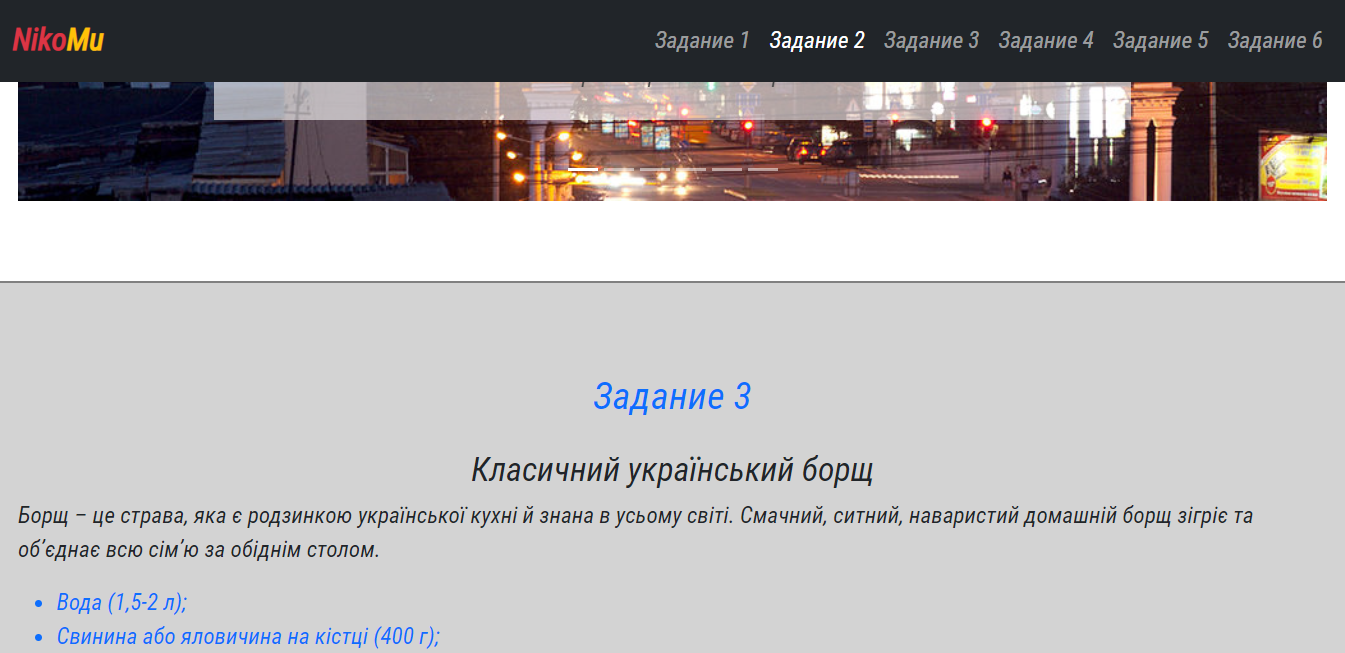


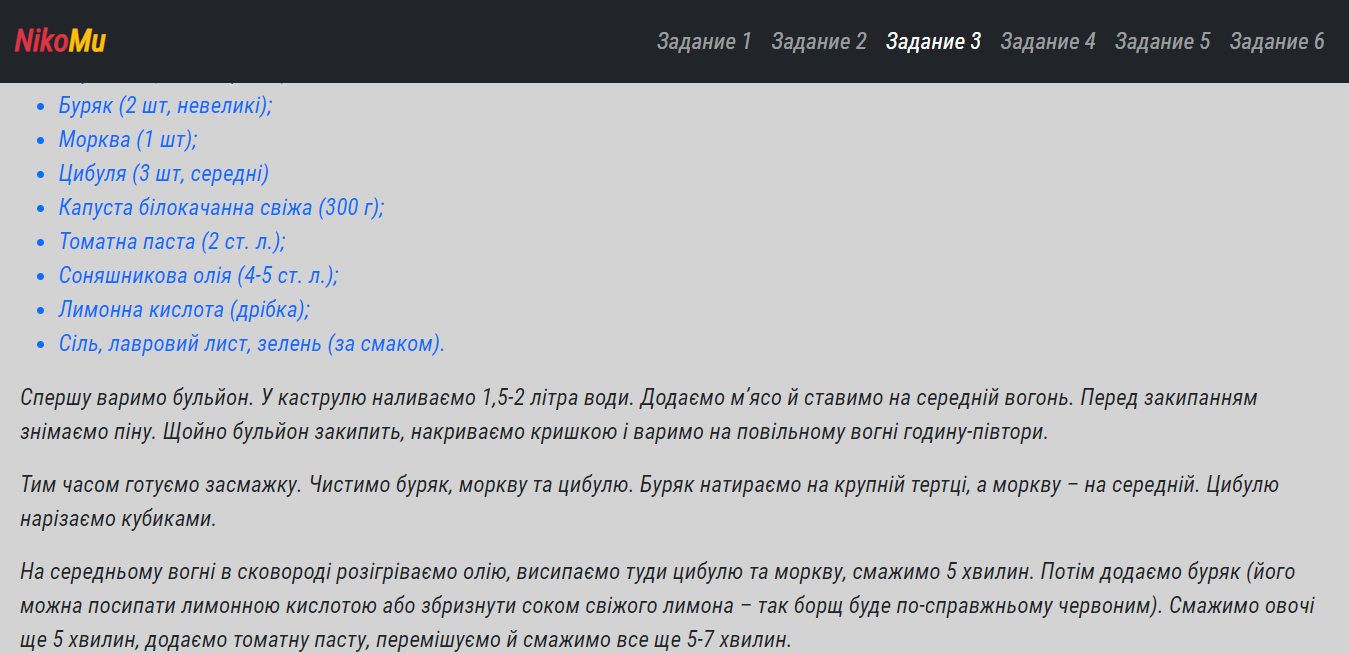
*Рис.1 - Запускаем сервер на Node.js*

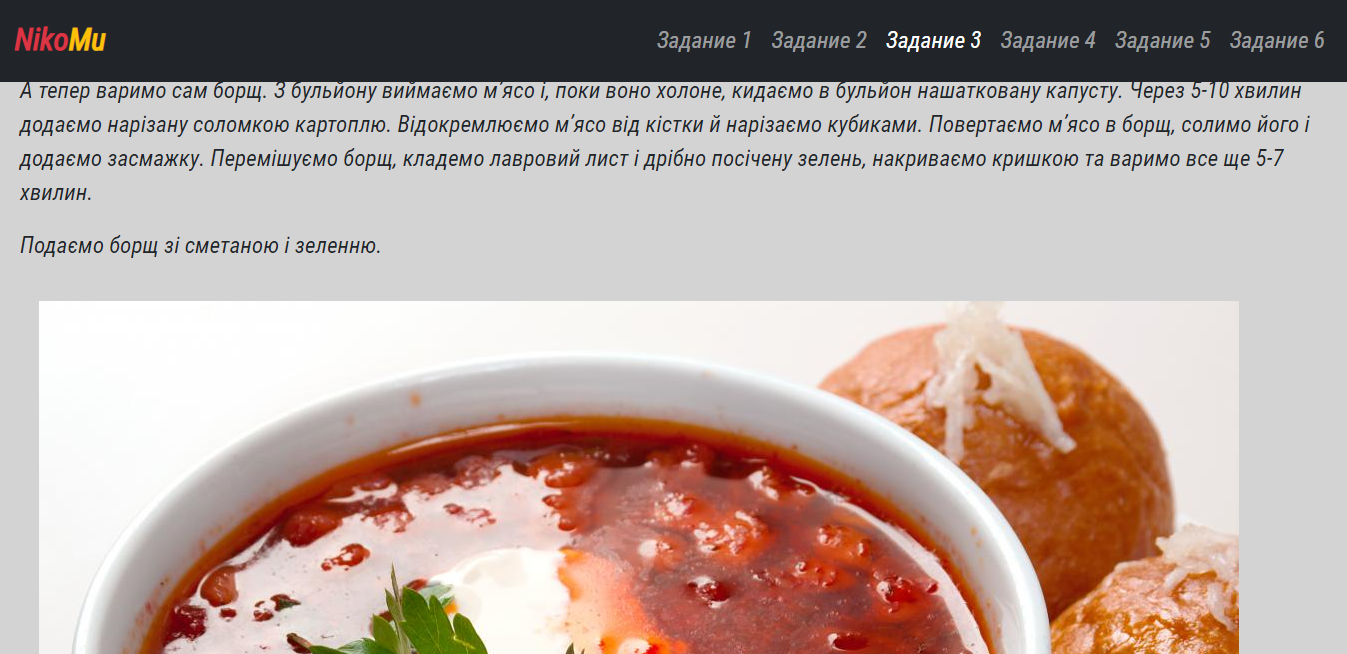


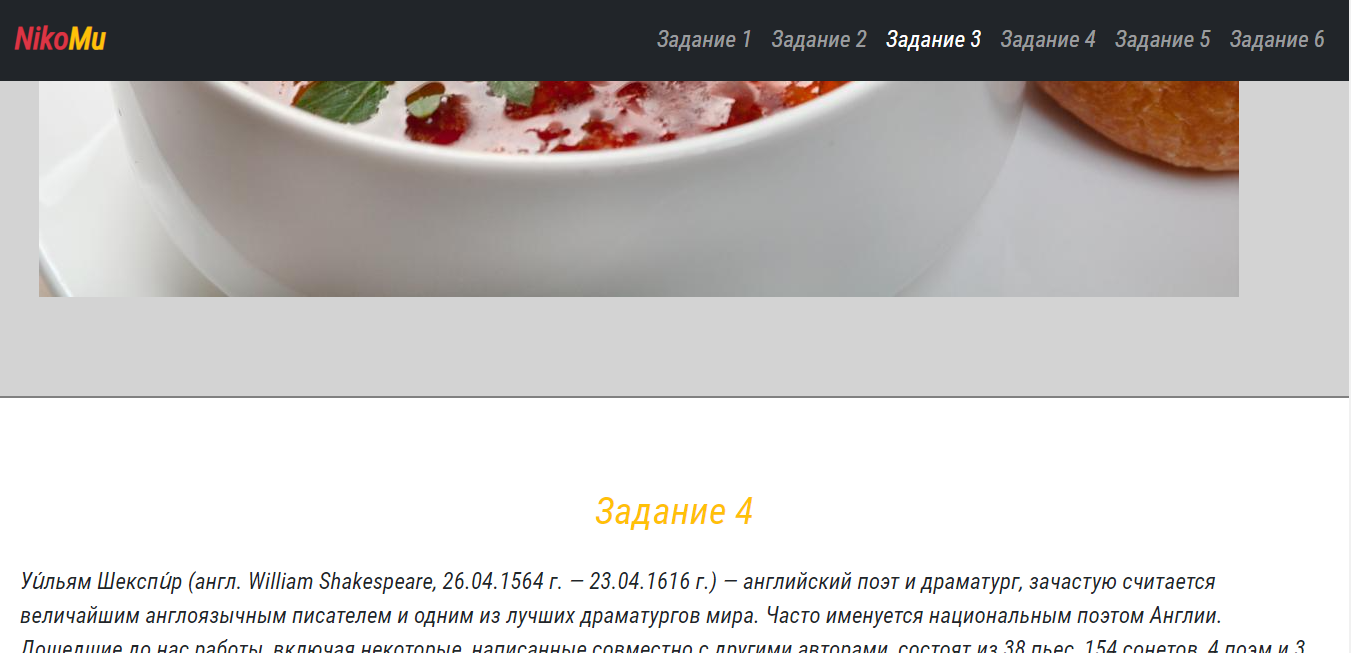


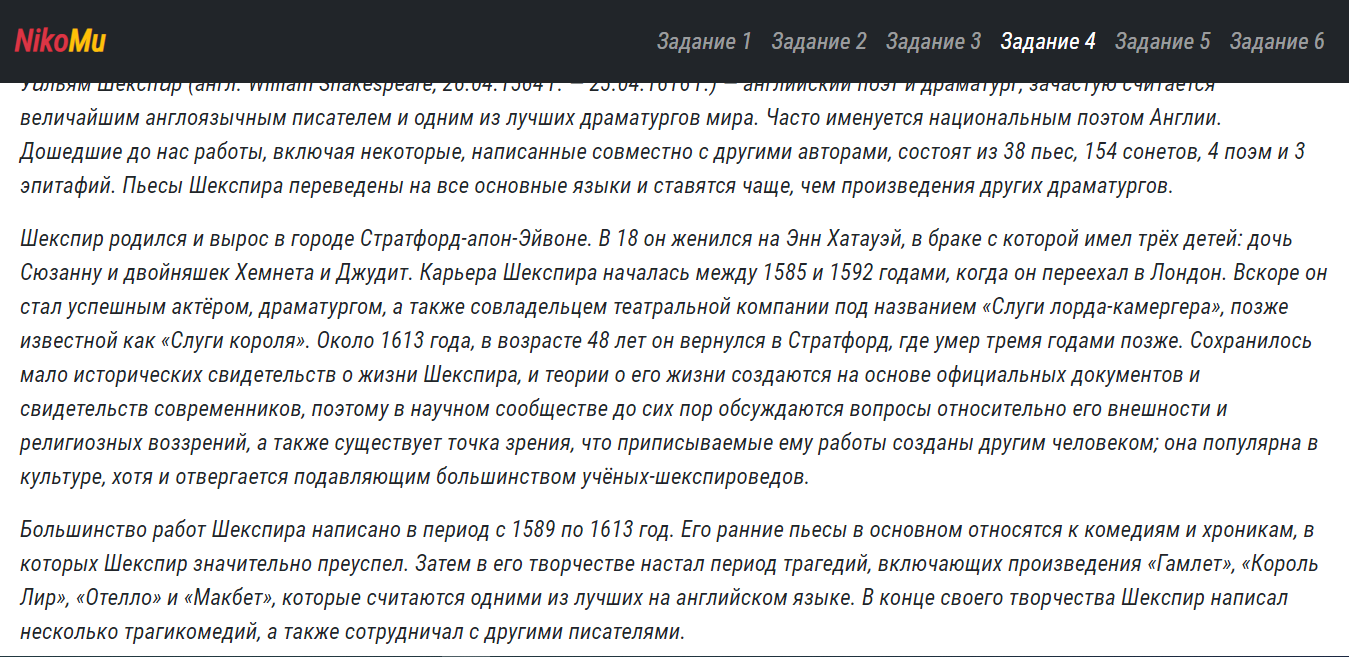


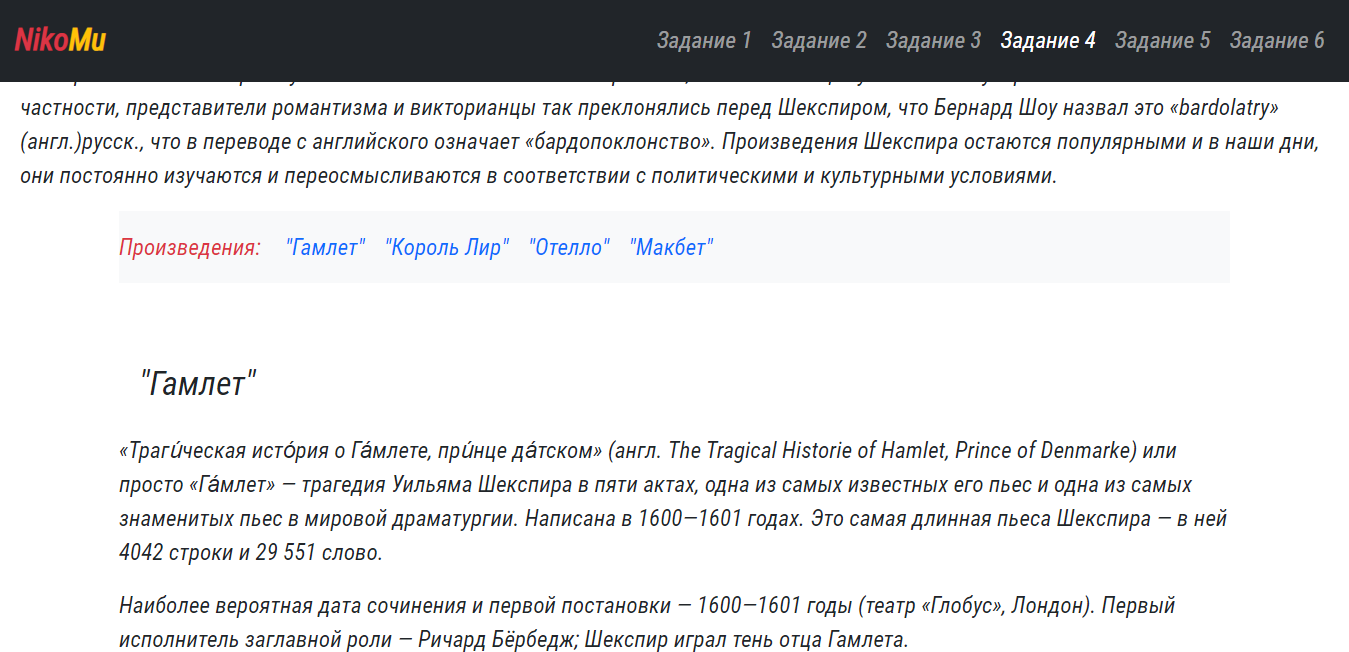


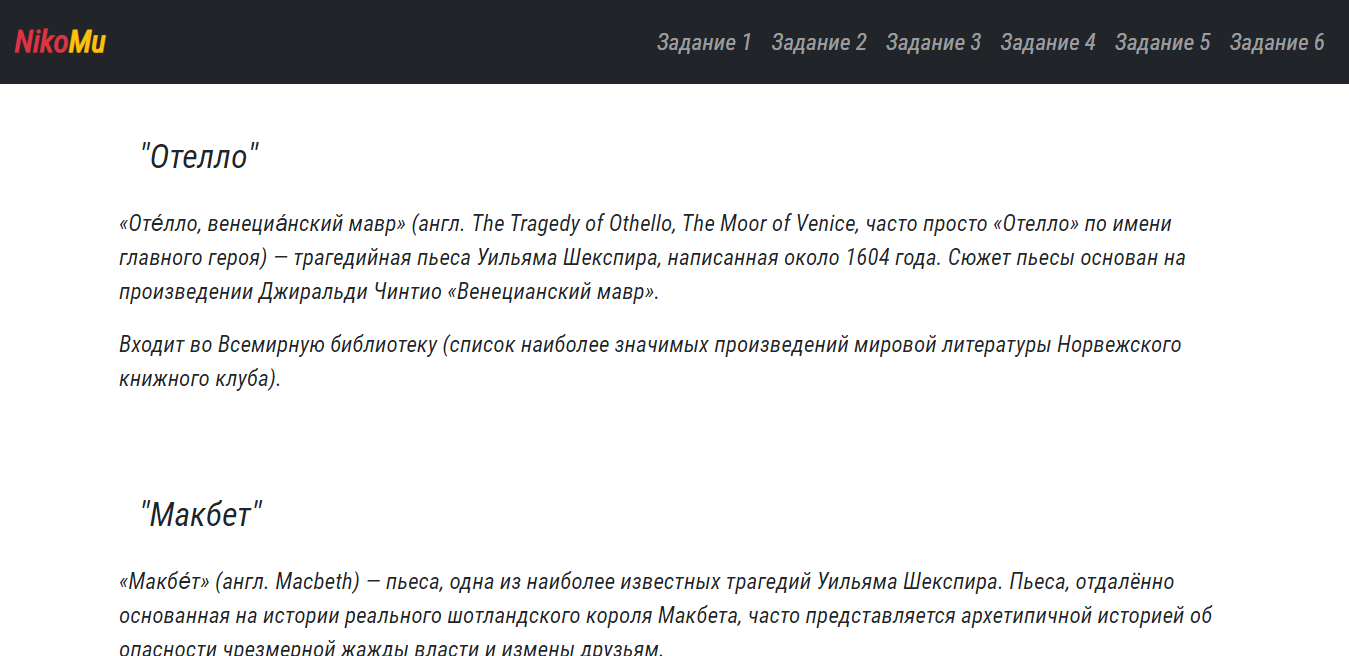


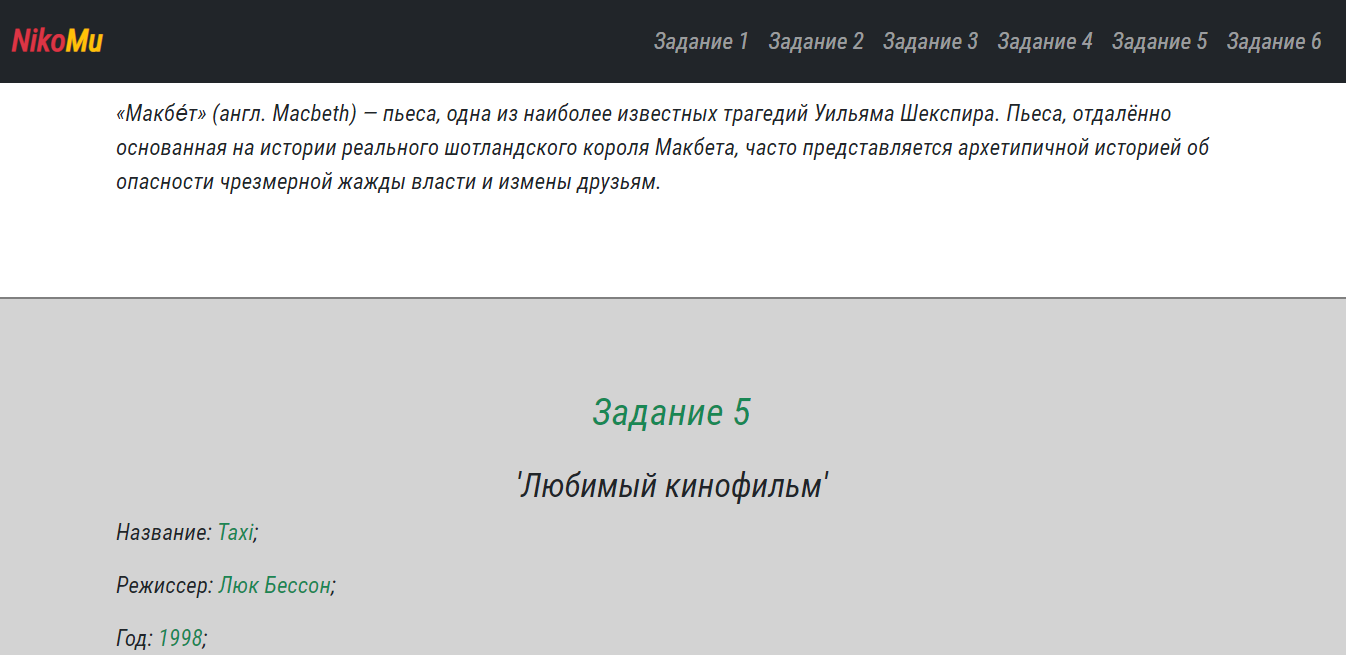




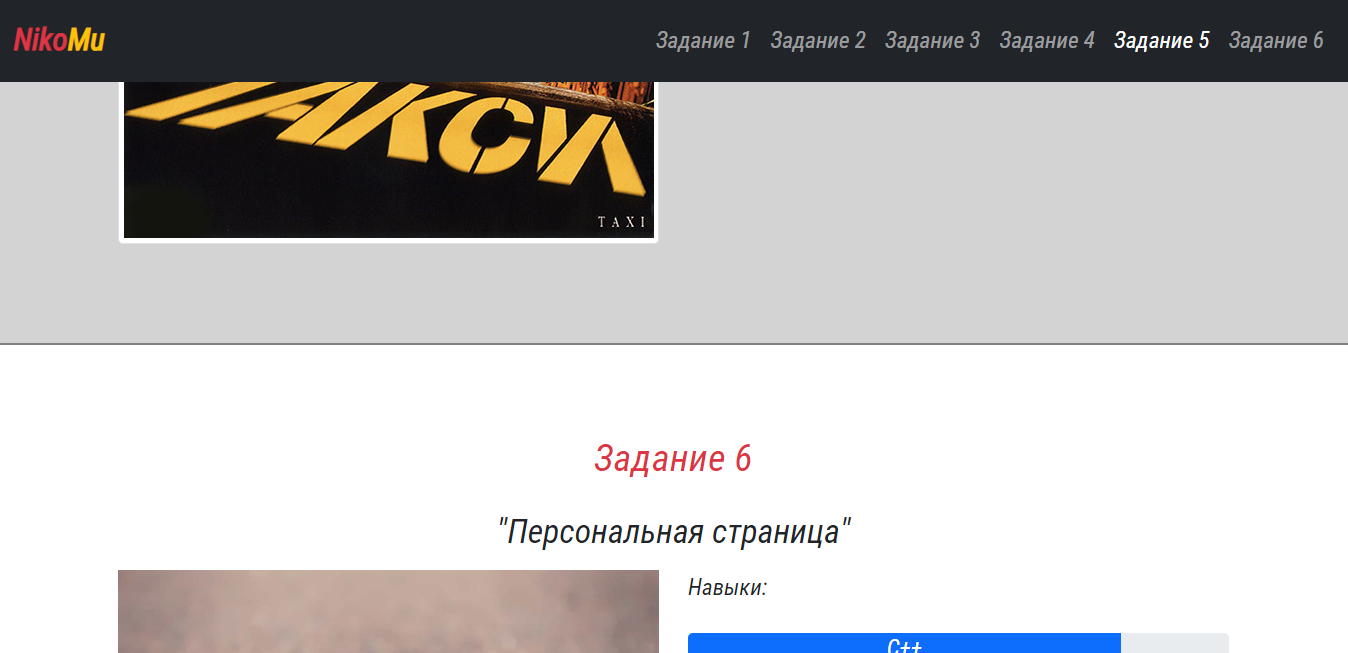


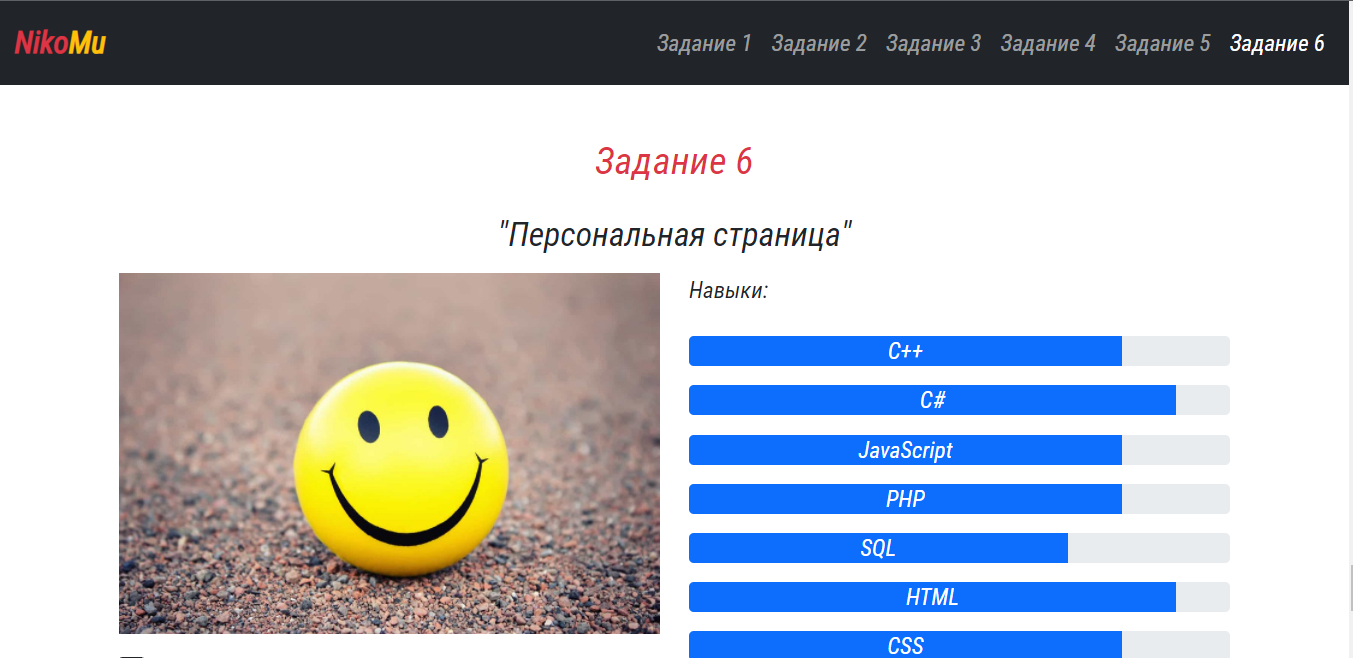


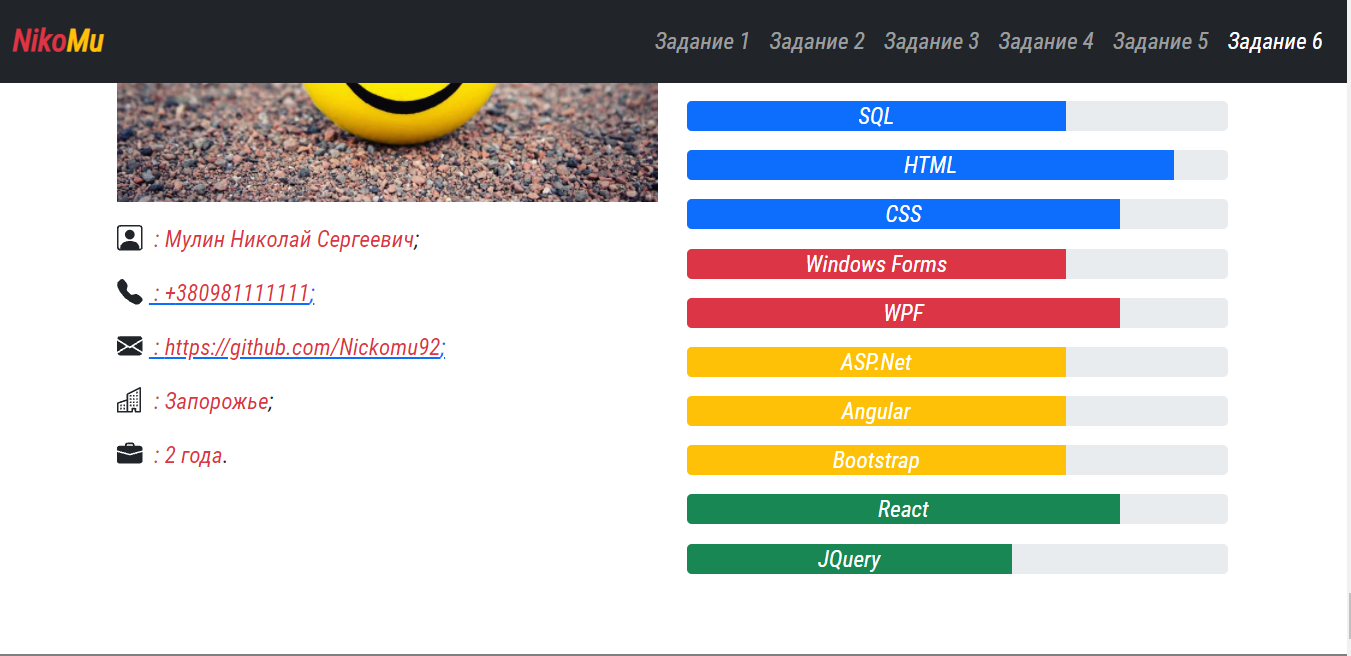












*Рис.2-16 – Общий вид сайта.*