**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

по дисциплине «**Современные средства *Front-end* разработки»**

на тему: «**Применение *JavaScript* для обработки форм и динамического управления *HTML*-документом**»

Выполнил: магистрант гр. МАГ 40-21

Вергейчик А.П.

Принял: преподаватель

Белявский Ю.В.

Гомель, 2019

**Цель работы:** изучить основные приёмы работы с *JavaScript* для обработки форм и динамического управления *HTML*-документом.

**Задание:** вариант задания выбирается по номеру студента в журнале.

В соответствии с вариантом, создать макет сайта заданной структуры (рисунок 1). Для оформления и позиционирования содержимого страницы использовать CSS.

Четные варианты

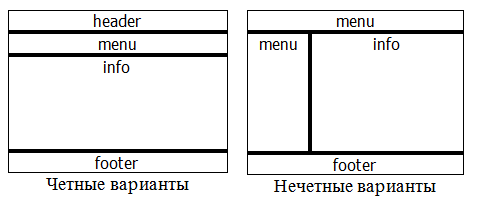


Рисунок 1 – Структура сайта

Макет страницы выполняется шириной – 800px, с выравниванием – по центру и содержит следующие основные блоки:

1. header – заголовок сайта, содержит логотип (изображение) и название;

2. menu – меню навигации (ссылки на задания 1 – 3);

3. info – основной раздел в котором располагается все информативная часть сайта;

4. footer – содержит ФИО студента, группу, дату создания, емайл и др.

Разрабатываемая WEB – страница, должна иметь следующую структуру:

1. Один HTML документ;

2. CSS файл (папка CSS);

3. Файл или файлы со скриптами (папка JS).

Все изменения вида и содержимого страницы, производятся динамически с помощью JavaScript, без загрузки других HTML документов (при необходимости они могут создаваться только динамически).

Задание 1. После загрузки страницы пользователю предлагается пройти регистрацию, для чего он должен заполнить следующие поля формы:

1. Фамилия – поле ввода;

2. Имя – поле ввода;

3. Пол – четные варианты список, не четные переключатель;

4. Возраст – поле ввода;

5. Логин – поле ввода;

6. Пароль – поле ввода с скрытием данных;

7. Е – майл;

8. Добавить 2 своих поля.

Форма должна содержать кнопки «Регистрация» и «Очистка полей». По нажатию на кнопку «Регистрация», необходимо проверить:

1. заполнение всех полей формы (обязательно для всех студентов);

2. валидность введенных данных (слабым можно не делать).

Пользователю необходимо сообщить, какие поля заполнены не верно или о успешном прохождении регистрации. Доступ к остальным элементам WEB-страницы, только после прохождения регистрации. Введенная информация сохраняется для дальнейшего использования.

Задание 2. После прохождения регистрации, форма скрывается. На странице выводится описание теста с предложением пройти его. В соответствии с вариантом (см. приложение), необходимо реализовать прохождение теста. Вопросы задаются последовательно. Для каждого вопроса динамически формируется новое окно (не диалоговое), в котором с помощью элементов форм – поле ввода, список, переключатель, выбирается вариант ответа. Также окно должно содержать кнопки «Следующий вопрос» и «Отмена». При нажатии на кнопку «Отмена» тест прекращается (возобновление только с начала).

После прохождения теста, заголовок теста скрывается, и выводятся данные о тестируемом, результат прохождения теста и время затраченное на его прохождение.

**Ход работы:**

Для удобства разработки вынесем вопросы и ответы в отдельный файл. Создадим структуру страницы согласно варианту. На рисунке 2 представлен общий вид созданной страницы.

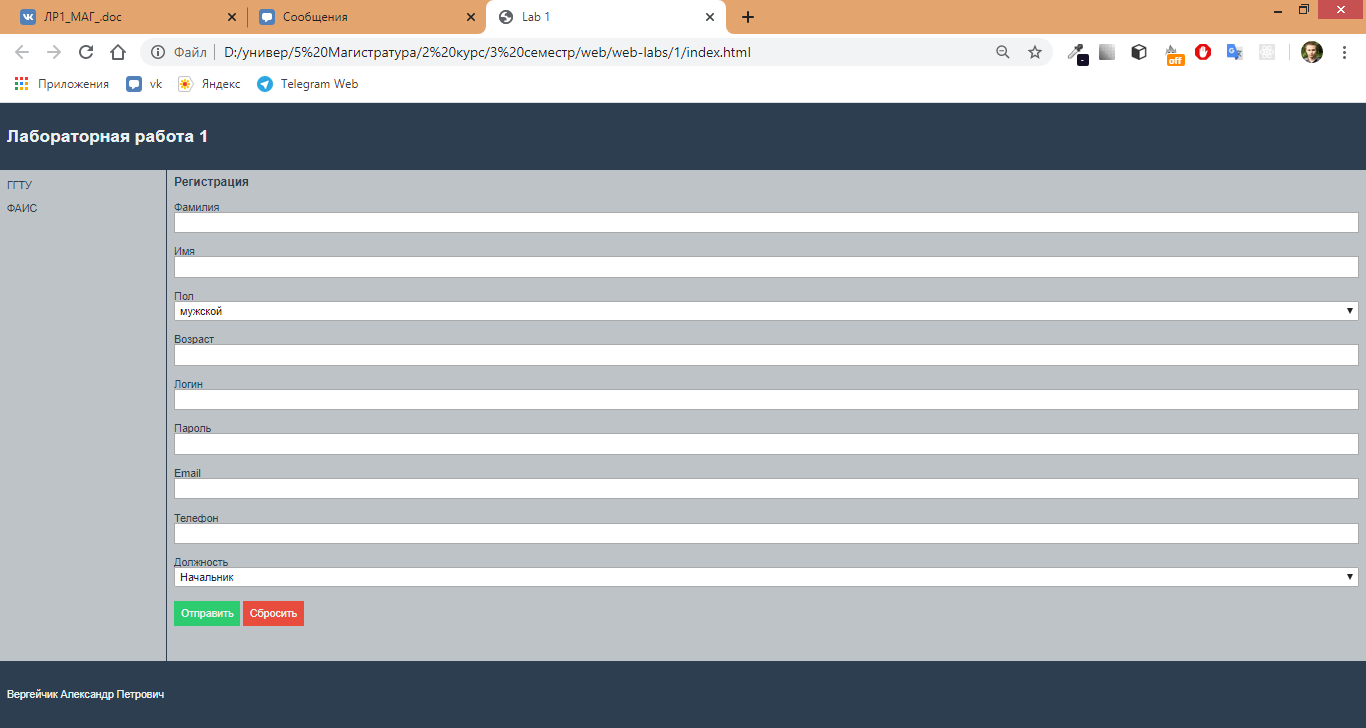


Рисунок 2 – Общий вид созданной страницы

Добавим верификацию. Для этого добавим в *HTML*-разметку тегов формы специальный атрибут, который помечает поля как обязательными для заполнения и проверяет их с помощью регулярных выражений на соответствие шаблону формата ввода. На рисунке 3 представлено сообщение об неверном заполнении поля.

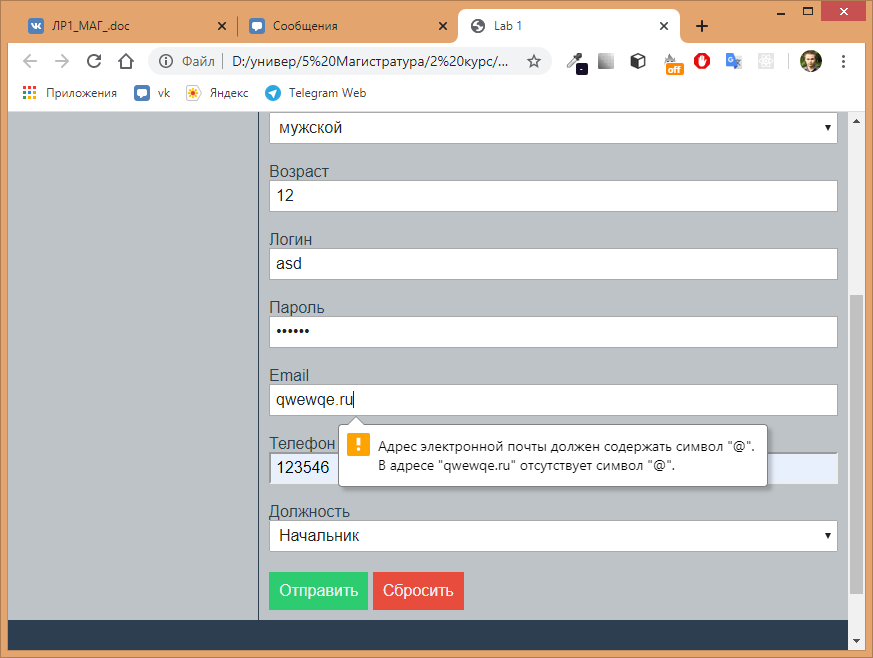


Рисунок 2 – Пример валидации данных

После отправки формы с введенной информацией, она будет получена с помощью *JavaScript* и также проверена на корректность заполнения всех полей. Если все поля заполнены, то перед пользователем появляется тест (рисунок 3).

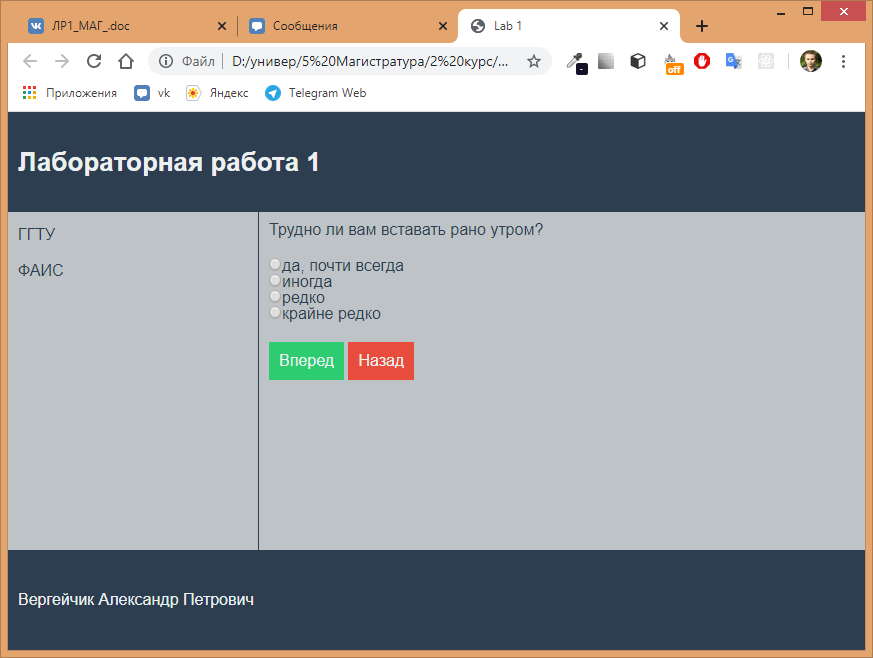


Рисунок 3 – Процесс прохождения теста

После прохождения на экран выводится подробная информация (рисунок 4).

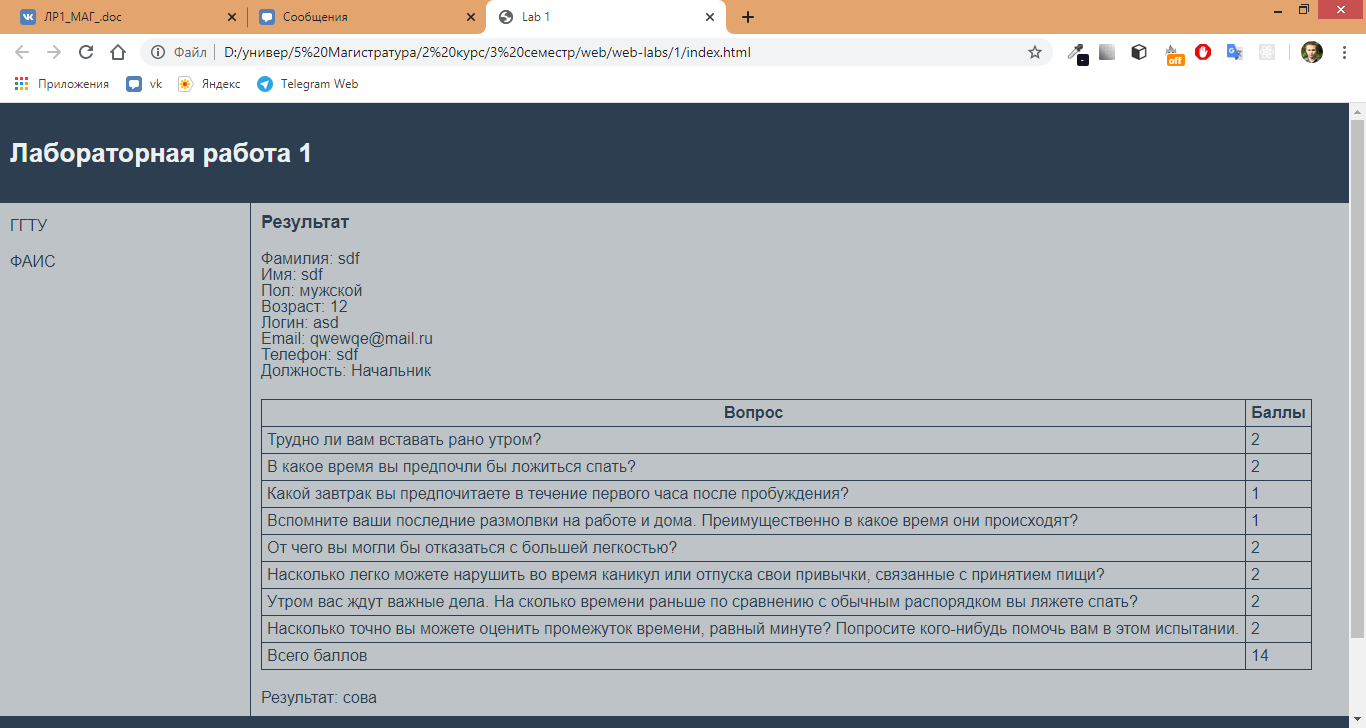


Рисунок 4 – Результат прохождения теста

**Вывод:** в ходе лабораторной работы было создано одностраничное приложение для прохождения тестирования. Все введенные пользователям данные проверяются на корректность. Подстановка необходимого представления осуществляется с помощью языка программирования *JavaScript*. После прохождения теста на экран выводится подробная информация.

**ПРИЛОЖЕНИЕ А**

(Обязательное)

**Исходный код *JavaScript*-файла**

let registrationBlock;

let testingBlock;

let resultBlock;

let user;

class User {

constructor(surname,

name,

gender,

age,

login,

password,

email,

phone,

position) {

this.surname = surname;

this.name = name;

this.gender = gender;

this.age = age;

this.login = login;

this.password = password,

this.email = email;

this.phone = phone;

this.position = position;

}

}

class Quiz {

constructor() {

this.i = 0;

this.answers = new Array(QUESTIONS.questions.length);

}

next() {

if (this.checkAnswer()) {

if (++this.i <= QUESTIONS.questions.length - 1) {

this.showQuiz();

} else {

this.showResult();

}

} else {

alert ('Error! Set answer');

}

}

prev() {

this.i--;

this.showQuiz();

}

checkAnswer() {

const answer = document.querySelector('input[name="answer"]:checked');

if (answer) {

this.answers[this.i] = Number(answer.value);

return true;

}

return false;

}

showQuiz() {

const question = document.querySelector('.question');

question.innerHTML = '<div class="question">' + QUESTIONS.questions[this.i].text + '<div>';

let fragment = document.createDocumentFragment();

QUESTIONS.questions[this.i].answers.forEach((answer) => {

const div = document.createElement('div');

const input = document.createElement('input');

input.type = 'radio';

input.value = answer.point;

input.name = 'answer';

const span = document.createElement('span');

span.innerHTML = answer.text;

div.appendChild(input);

div.appendChild(span);

fragment.appendChild(div);

});

question.appendChild(fragment);

}

showResult() {

let div = document.createElement('div');

div.classList.add('user-info');

div.innerHTML += `<div>Фамилия: ${ user.surname }</div>`;

div.innerHTML += `<div>Имя: ${ user.name }</div>`;

div.innerHTML += `<div>Пол: ${ user.gender }</div>`;

div.innerHTML += `<div>Возраст: ${ user.age }</div>`;

div.innerHTML += `<div>Логин: ${ user.login }</div>`;

div.innerHTML += `<div>Email: ${ user.email }</div>`;

div.innerHTML += `<div>Телефон: ${ user.surname }</div>`;

div.innerHTML += `<div>Должность: ${ user.position }</div>`;

let table = document.createElement('table');

table.classList.add('result\_\_table')

table.innerHTML = '<tr><th>Вопрос</th><th>Баллы</th></tr>';

QUESTIONS.questions.forEach((question, index) => {

table.innerHTML += `<tr><td>${ question.text }</td><td>${ this.answers[index] }</td></tr>`;

});

const resultSum = this.answers.reduce((sum, item) => {

return sum + item;

});

const resultText = QUESTIONS.results.filter((result) => {

return (resultSum >= result.min && resultSum <= result.max);

})[0].text;

table.innerHTML += `<tr><td>Всего баллов</td><td>${ resultSum }</td></tr>`;

resultBlock.appendChild(div);

resultBlock.appendChild(table);

resultBlock.innerHTML += 'Результат: ' + resultText;

testingBlock.style.display = 'none';

resultBlock.style.display = 'block';

}

}

document.addEventListener('DOMContentLoaded', () => {

registrationBlock = document.querySelector('#registration');

testingBlock = document.querySelector('#test');

resultBlock = document.querySelector('#result');

testingBlock.style.display = 'none';

resultBlock.style.display = 'none';

const registrationForm = document.querySelector('.registration-form');

registrationForm.addEventListener('submit', function(event) {

event.preventDefault();

registrationBlock.style.display = 'none';

testingBlock.style.display = 'block';

user = new User(

this.querySelector('#surname').value,

this.querySelector('#name').value,

this.querySelector('#gender').value,

this.querySelector('#age').value,

this.querySelector('#login').value,

this.querySelector('#password').value,

this.querySelector('#email').value,

this.querySelector('#phone').value,

this.querySelector('#position').value

);

let test = new Quiz();

const answerNextButton = document.querySelector('#answer\_next');

const answerPrevButton = document.querySelector('#answer\_prev');

answerNextButton.addEventListener('click', () => {

test.next();

});

answerPrevButton.addEventListener('click', () => {

test.prev();

});

test.showQuiz();

});

});