Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Санкт-Петербургский государственный технологический институт

(технический университет)»

(СПбГТИ(ТУ))

Факультет информационных технологий и управления

Кафедра систем автоматизированного проектирования и управления

Дисциплина: Средства визуализации данных

**Лабораторная работа № 5**

**«Unit тесты для программного комплекса»**

Студенты 475 группы:

Г.Р. Ананенков

М.А. Видинеев

П.А. Петренко

Е.А. Поторочин

Преподаватель:

В. А. Соболевский

Санкт-Петербург

2020

1. **Введение**
   1. **Назначение**

Назначением данного документа является создание и описание Unit тестов для трех Use кейсов, указанных в Use Case Speciﬁcation.

* 1. **Область применения**

Данный программный комплекс предназначен для предоставления комплекса тренировок по, заданным пользователем характеристикам и предпочтениям.

1. **Прецеденты** 
   * 1. **Описание прецедента**

Пользователь использует приложение для одноразовой тренировки без регистрации. Для этого требуется выбрать одну из предложенных тренировок на главной странице сайта.

Стек технологий:

Язык: Python 3.6

Фреймворк: Robot framework

Библиотека: SeleniumLibrary

* + 1. **Назначение тестируемого модуля (функции)**

Назначение теста состоит в том, чтобы проверить, получит ли пользователь тренировку, перейдя к ней на главной странице сайта.

* + 1. **Демонстрация прохождения теста**

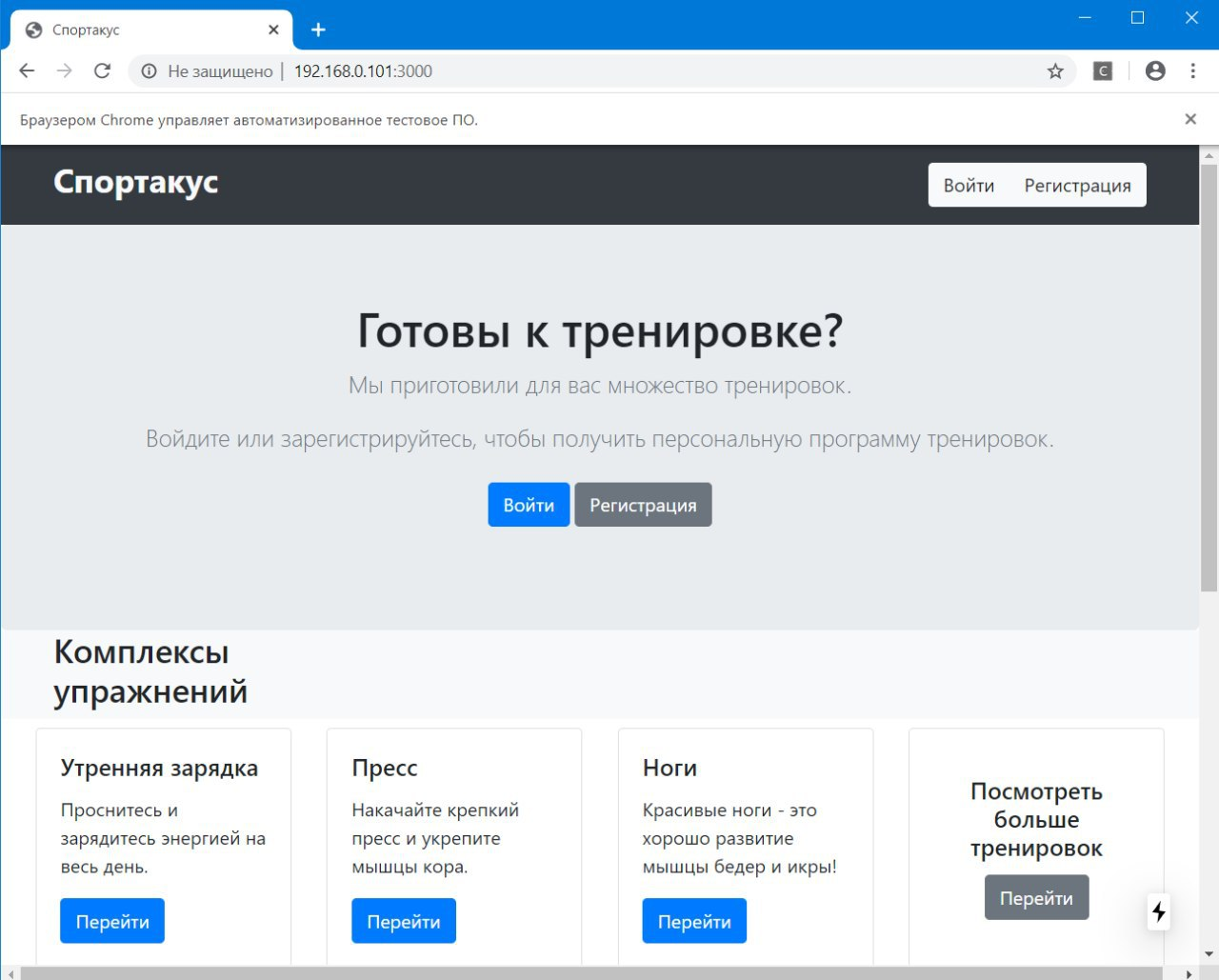
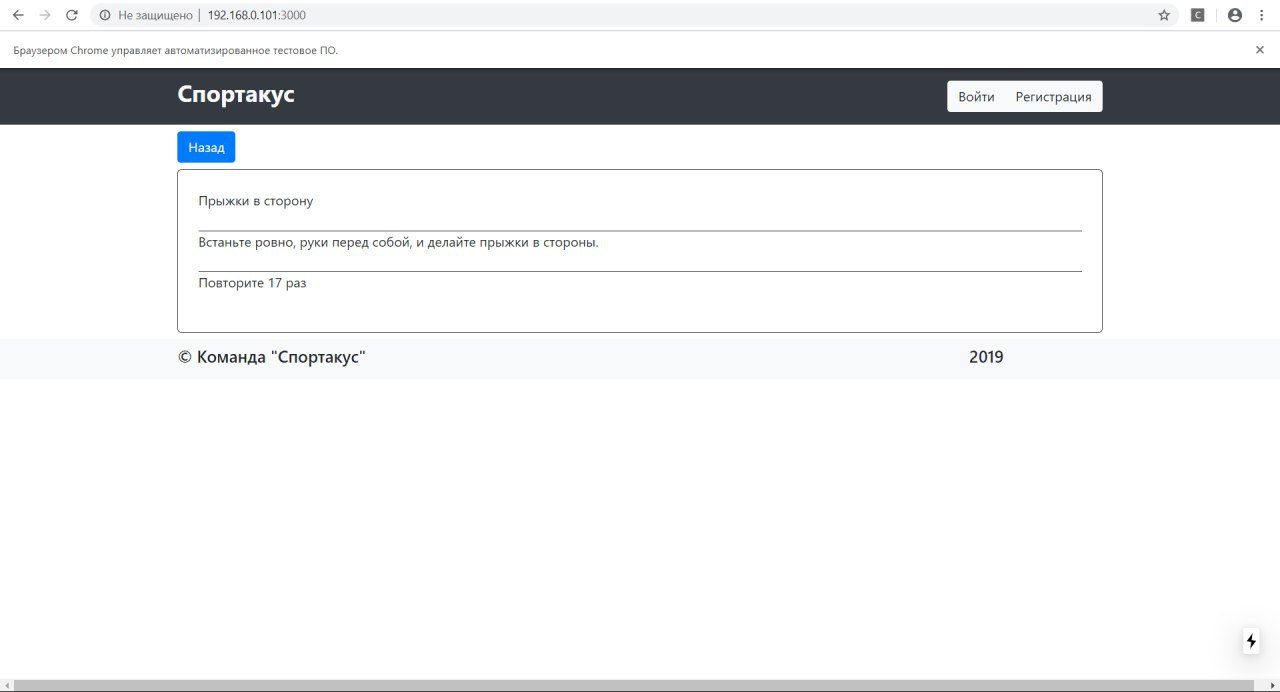


Рисунок 1 – Начало тестирования

Рисунок 2 – Получение результата

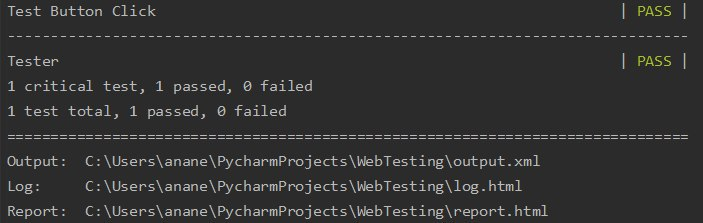


Рисунок 3 – Результат в консоли

* + 1. **Listing**

\*\*\* Settings \*\*\*

Library SeleniumLibrary

\*\*\* Variables \*\*\*

${Browser} Chrome

${URL} http://192.168.0.101:3000

\*\*\* Test Cases \*\*\*

Test Button Click

Open Browser ${URL} ${Browser}

Maximize Browser Window

Click Button name:exer-button-0

* + 1. **Описание прецедента**

Пользователь хочет использовать фитнес приложение на постоянной основе, поэтому ему необходимо будет зарегистрироваться. На вкладке “Построить тренировку”, пользователем вводятся настройки для получения наиболее подходящего комплекса тренировок.

Стек технологий:

язык: JavaScript

библиотека Request

* + 1. **Назначение тестируемого модуля (функции)**

Назначение теста состоит в том, чтобы программа выдала комплекс тренировок по заданным характеристикам.

* + 1. **Демонстрация прохождения теста**

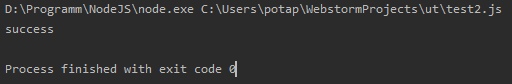


Рисунок 4 – Результат в консоли

* + 1. **Listing**

//функция осуществляющая GET запрос

//конкретный модуль

function reqGet(url) {

//возвращает обьект, в котором содержится

//либо ответ, в случае успешного выполнения запроса

//либо ошибка, в случае выполнения с ошибкой

return new Promise(function (resolve, reject) {

//загрузка модуля 'request' для удобной работы

//с http запросами

let request = require("request");

//формирование параметров запроса

const options = {

url: "http://25.78.98.95:8080/api/v1/userModules/userModules",

method: "GET"

};

request(options.url + encodeURI(url), function (error, response, body) {

//при ошибке передаем для возврата ошибку

//при успешном выполнении запроса передаем

//то, что получили в ответ на запрос

if(error) reject(error);

else{

resolve(body);

}

});

})

}

//функция осуществляющая GET запрос

//которая принимает параметры: url - вызывающий

//конкретный модуль и postData - данные

//в фромате JSON для передачи серверу в запросе.

//возвращает обьект, в котором содержится

//либо ответ, в случае успешного выполнения запроса

//либо ошибка, в случае выполнения с ошибкой

function reqPost(url, postData){

return new Promise(function (resolve, reject) {

//загрузка модуля 'request' для удобной работы

//с http запросами

let request = require("request");

//формирование параметров запроса

const options = {

url: "http://25.78.98.95:8080/api/v1/userModules/userModules",

method: "POST"

};

request.post({url: options.url + encodeURI(url), form: postData}, function (error, response, body) {

//при ошибке передаем для возврата ошибку

//при успешном выполнении запроса передаем

//то, что получили в ответ на запрос

if(error) reject(error);

else{

resolve(body);

}

});

})

}

//export функций reqGet и reqPost

//чтобы их можно было вызывать из других модулей

module.exports.reqGet = reqGet;

module.exports.reqPost = reqPost;

* + 1. **Описание прецедента**

Системный администратор заносит новые программные комплексы или изменяет существующие данные в базе данных.

Стек технологий:

язык: JavaScript

библиотека Request

* + 1. **Назначение тестируемого модуля (функции)**

Назначение теста заключается в проверке добавления новых комплексов тренировок.

* + 1. **Демонстрация прохождения теста**

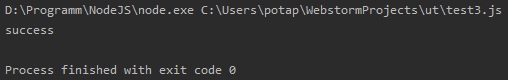


Рисунок 5 – Результат в консоли

* + 1. **Listing**

//подключение функций из модуля req.js

let req = require('./req.js');

//переменная в которой хранится состояние теста "пройден"/"не пройден"

let isFailed = false;

//запрос на добавление пользователя

req.reqGet("/addUser/testLogin/testPass").then(response =>{

//обработка ответа

let json = JSON.parse(response);

//если код ответа не 200 (код, который возвращается

// при удачном запросе), тогда меняем

//статус на "не пройден"

if(json.code !== 200) isFailed = true;

//выполняем запрос на авторицию

return req.reqGet("/signIn/testLogin/testPass");

}).then(response =>{

let json = JSON.parse(response);

if(json.code !== 200) isFailed = true;

//запрос на обновление данных пользователя

return req.reqGet("/setUserData/testLogin/Мужской/180/80/Средне");

}).then(response =>{

let json = JSON.parse(response);

if(json.code !== 200) isFailed = true;

//запрос на получение комплекса

//упражненй для пользователя

return req.reqGet("/getUserComplex/testLogin/Мужской/80/Средне/Ягодицы");

}).then(response =>{

let json = JSON.parse(response);

if(json.code !== 200) isFailed = true;

//в случае если какая-либо

//из функций отработала

//с ошибкой, меняем статус

//теста на "не пройден"

}).catch(error => {

isFailed = true;

}).then(final =>{

//вывод в консоль, в зависимости от статуса

//теста, соответствующего сообщения

if(isFailed) console.log("fail");

else console.log("success");

});

//подключение функций из модуля req.js

let req = require('./req.js');

//переменная в которой хранится состояние теста "пройден"/"не пройден"

let isFailed = false;

//вспомогательная переменная для хранения id компекса

let complexId;

//выполнение запроса на авторицию администратора

req.reqGet("/signInAdmin/admin/admin").then(response =>{

//обработка ответа

//если код ответа не 200 (код, который возвращается

//при удачном запросе), тогда меняем

//статус на "не пройден"

let json = JSON.parse(response);

if(json.code !== 200) isFailed = true;

//формирование JSON-обьекта для передачи

//в POST запрос для добавления комплекса

let complexBody ={

name: "testName",

description: "testDescription",

category: "Спина",

sex: "Мужской",

difficult: "Легко",

minWeight: "66",

maxWeight: "99"

};

//выполнение запроса на добавление комплекса

//в ответе приходит id созданного комплекса

return req.reqPost("/addComplex/", complexBody);

}).then(response =>{

let json = JSON.parse(response);

if(json.code !== 200) isFailed = true;

//добавляем значение переменной complexId

//для использования далее

complexId = json.id;

//формирование JSON-обьекта для передачи

//в POST запрос для изменения комплекса

//в нем присутсутвует id

let complexBody ={

complexId: json.id,

name: "testNameUpdate",

description: "testDescriptionUpdate",

category: "Спина",

sex: "Мужской",

difficult: "Легко",

minWeight: "33",

maxWeight: "66"

};

//выполнение запроса на

//изменение комлекса

return req.reqPost("/updateComplexById/", complexBody);

}).then(response =>{

let json = JSON.parse(response);

if(json.code !== 200) isFailed = true;

//выполнеие запроса на удаление

//созданного и измененного комплекса

return req.reqGet("/deleteComplexById/" + complexId);

}).then(response =>{

let json = JSON.parse(response);

if(json.code !== 200) isFailed = true;

//в случае если какая-либо

//из функций отработала

//с ошибкой, меняем статус

//теста на "не пройден"

}).catch(error => {

isFailed = true;

}).then(final =>{

//вывод в консоль, в зависимости от статуса

//теста, соответствующего сообщения

if(isFailed) console.log("fail");

else console.log("success");

});

1. **Вывод**

Была проведена проверка всех Use Case, которая подтвердила работоспособность всех основных систем программного комплекса.