STUDENT USE		STAFF USE	
Module Name	Web Technology	First Marker's (acts as signature)	
Module Code	4BUIS011C	Second Marker's (acts as signature)	
Lecturer Name	Olga Yugay	Agreed Mark	
UoW Student IDs		**For Registrar's office use only (hard copy submission)**	
WIUT Student IDs	00007242		
Deadline date	15/11/2018		
Assignment Type	☐Group☑Individual		

COURSEWORK SUBMISSION FORM

MARKERS FEEDBACK (Continued on the next page)

1. Introduction

The website that I have created for this coursework is an online-job searching and posting platform. I took design inspiration and content from the official website of SimplyHired, https://simplyhired.com. The following tools have been used during the development process of the website:

1. Bootstrap v4.1.3
2. jQuery 3.3.1
3. Font Awesome 5.5.0

2. Audience Profile

The website targets the audience based on the following factors.

2.1. Demographics

As the website has been created in English, the audience is mostly English-speaking people. It covers both genders in the age group of 18 to 35.

In terms of location, the main users of the website are expected to be from first-tier, English-speaking countries like USA, UK and Australia.

2.2. Interests

The people who use a job searching platform are usually either unemployed or looking forward to change their current job.

This platform can also be used by companies and other organizations to recruit new employees and contractors.

3. Features and Functionality

3.1. Features

One of the best features of the website that I have created is that it is responsive. It keeps its efficiency and design in all kind of device screens from mobiles to large desktop screens.

It has got a clear navigation, which makes it easy to browse around the website. Because there is only 7 pages in the website, users can get to where they want in a matter of one click.

The website is hosted on the Internet. It is online at https://simplyhired2018.firebaseapp.com.

3.2. Functionality

The main function of the website is to search for available vacancies based on job titles that user enters. Although there is no database to support such kind of functionality, I created a JSON file myself and used it as a database. The file, data.json, is an array of objects where each object is an instance of jobs. It stores job title, salary and location information.

User can search for jobs using the search form located in the first section of the home page. The logic behind it uses search.js file. It takes the input of the user in the search form and looks for the object that contains that keyword as a job title value inside data.json file.

```
// Searching from json

$(document).ready(function(){
  $.ajaxSetup({ cache: false });
  $('#search').keyup(function(){
   $('#result').html('');
   $('#state').val('');

  //  Getting the input of the user in the search field
   var searchField = $('#search').val();
   var expression = new RegExp(searchField, "i");

  //  getting data form data.json file
   $.getJSON('data.json', function(data) {
    $.each(data, function(key, value){
     if (value.title.search(expression) != -1 || value.location.search(expression) != -1)
     {
      // dispaying the results to the user
      $('#result').append('<li class="list-group-item link-class"> <strong>'+ value.title
      +'</strong>| '+ value.salary +' | <span class="text-muted">'+value.location
      +'</span></li>');
     }
    });
   });
  });

  $('#result').on('click', 'li', function() {
   var click_text = $(this).text().split('|');
   $('#search').val($.trim(click_text[0]));
   $("#result").html('');
  });
 });
```

Users can also post available vacancies. In order to do so, they need to go to jobpost.html page. On this page there are multiple buttons that pops up the modal with a job posting form inside. Modal form was created using Bootstrap. The form asks for company name, available job position, job description, where the job is located, salary and whether the job is part-time or full-time.

I get the “post” button and the form by their ID and add an eventListener when “post” button is clicked. The function gets executed on button-click. It gets all the inputs of the user and creates a “job” object that stores all of that information. Then, this object is pushed to “jobs” array, which is stored in local storage for later use. Lastly, it alerts the user to let know that form is submitted successfully.

```
var button = document.getElementById('submit');
var form = document.getElementById('postJob');

// Creating jobs array for saving data
var jobs = []

button.addEventListener('click', (e) => {
  e.preventDefault()

  // Getting values of users inputs
  let companyName = document.getElementById('companyName').value || undefined
  let jobTitle = document.getElementById('jobTitle').value || undefined
  let description = document.getElementById('description').value || undefined
  let location = document.getElementById('location').value || undefined
  let salary = document.getElementById('salary').value || undefined
  let x = document.getElementById("time").selectedIndex;
  let time = document.getElementsByTagName("option")[x].value || undefined;

  // creating an object
  let job = {
      company: companyName,
      title: jobTitle,
      description: description,
      location: location,
      salary: salary,
      time: time
  }

  // pushing the object to jobs array
  jobs.push(job);


  // Stores the JavaScript object as a string
  localStorage.setItem('jobs', JSON.stringify(jobs));

  alert("You have succussfull posted your job!");

  form.reset();
})
```

I displayed all the data that I collect from job postings on a table in a different page, localjobs.html. I simply accessed the data that is stored in local storage and display as a table. This function gets executes when the window is loaded. If the function cannot find any data it displays an alert saying “No data was entered”.

```
var $ = (arg) => document.querySelector(arg)
window.onload = () => {
  let tbody = $('#tbody')
  let jobs_string = localStorage.getItem('jobs')
  let jobs;

  if (jobs_string == null) {
    alert('No data was entered!')
  } else {
    jobs = JSON.parse(jobs_string)
  }

  for (job of jobs || []) {
    let body_content = `
      <tr>
        <td>${job.company}</td>
        <td>${job.title}</td>
        <td>${job.description}</td>
        <td>${job.location}</td>
        <td>${job.salary}</td>
        <td>${job.time}</td>
      </tr>
    `

    tbody.innerHTML += body_content
  }
}
```

One more function of the website is that it has a blog which displays different blog posts for job seekers and recruiters.

There are two buttons which act as tab menus and when one is clicked it shows the corresponding blog posts.

```
function showBlog(evt, blogname) {
  var i, tabcontent, tablinks;
  tabcontent = document.getElementsByClassName("tabcontent");
  for (i = 0; i < tabcontent.length; i++) {
      tabcontent[i].style.display = "none";
  }
  tablinks = document.getElementsByClassName("tablinks");
  for (i = 0; i < tablinks.length; i++) {
      tablinks[i].className = tablinks[i].className.replace(" active", "");
  }
  document.getElementById(blogname).style.display = "block";
  evt.currentTarget.className += " active";
}
```

References:

SimplyHired, (2018). Job Search Engine. [online] SimplyHired. Available from: https://simplyhired.com [Accessed at 10 November 2018].