University of Engineering and Technology

Department of Computer Engineering

Final term project

Field Finder-A Web Application

Submitted by:

Zainab Rizwan (2019-CE-36)

Ushna Ijaz (2019-CE-39)

Submitted to

Ma'am Sahar Waqar

Contents

I. Project description

1. Idea Name

Field Finder

2. Members List

1) Zainab Rizwan [2019-CE-36]

2) Ushna Ijaz [2019-CE-39]

3. Description

Field Finder is a web application that lets you search for Majors and Institutions offered nationwide. Each of these paths to knowledge opens up a world of opportunity. So this may help someone in finding their passion. We have a feature "Major Quiz" from which a user will answer the questions and based on the answers, majors will be suggested. This will help them choose a career path. We have a another feature "Merit Calculator" that lets you calculate your merit for different universities in Pakistan. This can help narrowing down options.

4. UI Screens and their description

Home Page

The homepage consists of a general overview of the appplication, showcasing the additional features included. Here are all the elements included the home page:

- Logo
- Navigation
- Headline
- Footer
- Map
- Contact Us form
- Overview of Features

Majors

Consists of lists of fields of studies with their majors. A user can easily browse through over 500 majors offered nationwide.

Institutions

Consists of a database of over 1500 institutions of Pakistan along with their location.

Merit Calculator

Contains merit calculators so students can calculate their aggregate for different universities around Pakistan.

Quiz

This is part where a student can take a quiz and through that it will recommend majors that fits that particular person.

Registration

A registration and login page was created that will appear before the quiz. A user will have to register in order to take the quiz.

II. Data structures

1. Data structures used and reason

Data Structures used:

We used lists for our quiz page. It was used to create the questions and the majors. We used list of dictionaries. This made it easier and more efficient for us since we could easily append or access from any index.

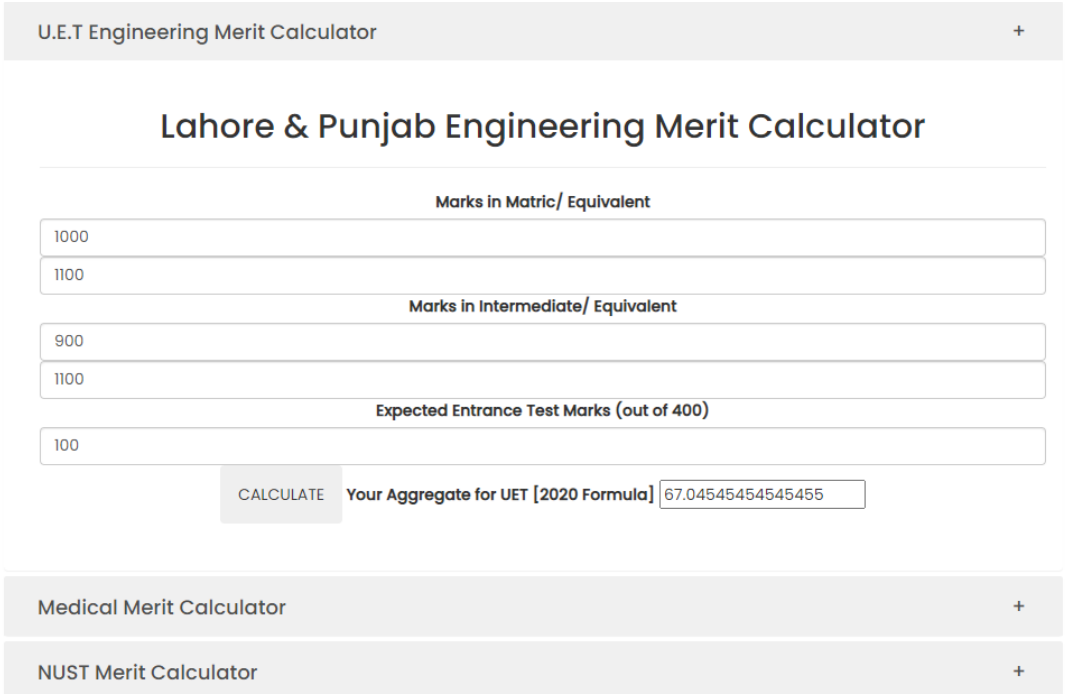
2. Algorithms used

Merit Calculator

We made some efficient algorithms for our web application. The merit calculator was made by the following functions:

```
 //Nust merit
var m_3, m_4, i_o, i_t, e_2, agg2;
function myFunction3()
{
 m_3=document.getElementById("m_3").value;
 m_4=document.getElementById("m_4").value;
 i_3=document.getElementById("i_3").value;
 i_4=document.getElementById("i_4").value;
 e_2=document.getElementById("e_2").value;
 agg2=((15*i_3)/i_4)+((75*e_2)/200)+((10*m_3)/m_4);
 document.getElementById("agg2").value = agg2; return false;
 }
```

This is an example of calculating the aggregate for U.E.T. The output that it shows on the webpage:



For the Login/SignUp form, we designed an algorithm like this:

Registration:

In [ ]:

```
get user Id (Name, password, email, phone etc.)
If (id data validated in front)
    Then send for backend validation
Else
    Go back to 1
If (backend validation OK)
    Then user creation OK
Else
    opération failure
```

Login

In [ ]:

```
get (mail or username) and password
If (both match with the ones in the database)
    Then allow login.
ELse
    Login failure
```

Quiz

For shuffling the questions we made it with the help of Durstenfeld shuffle algorithm. The time complexity is O (n), the space complexity is O (1). Help was taken from stackoverflow.

```
        */
    function shuffle(a) {
        var j, x, i;
        for (i = a.length; i; i -= 1) {
            j = Math.floor(Math.random() * i);
            x = a[i - 1];
            a[i - 1] = a[j];
            a[j] = x;
        }
    }
}();
```

Following functions were created:

1) checkAnswer:

This function was created for the option yes or no. If the user clicks yes it will take it as a yes or otherwise no, and then will move to the next question.

```
function checkAnswer( evt )
{
    // answer question with a yes or no
    if (evt.target.id === "quiz--yes") {
        answerQuestion( currentQuestionIndex, 'yes' );
    }
    else {
        answerQuestion( currentQuestionIndex, 'no' );
    }
    goToNextQuestion();
}
```

2) function answerQuestion:

This function will validate inputs. If the answer/question provided isn't in the list of dictionaries then the console will give an error.

```
function answerQuestion( i, answer )
{
    // validate inputs
    if ( !questions[i] ) {
        console.error('Question with index', i, 'not found');
        return;
    }
    if ( answer !== 'yes' && answer !== 'no' ) {
        console.error('Invalid answer', answer ,'provided to question with index', i, ':',
        questions[i].text);
        return;
    }
```

3) goToQuestion:

This function will update the html to go the next question. i is the index of question the var questions[].

```
function goToQuestion( i )
{
    if (currentQuestionIndex === i) {
        console.error('Attempting to transition to current question');
        return;
    }
    if ( !questions[i] ) {
        console.error('Question with index', i, 'not found');
        return;
    }

    questionContainer.classList.add('quiz__question--transitioning');
    setTimeout(function()
    {
```

4) goToNextQuestion and goToPrevQuestion

```
function goToNextQuestion()
{
    if (currentQuestionIndex < questions.length - 1) {
        goToQuestion( currentQuestionIndex + 1 );
    }
    else {
        goToResults();
    }
}

function goToPrevQuestion()
{
    goToQuestion( currentQuestionIndex - 1 );
}
```

Institutions For our institutions section, we used extracted data from our database table.

```
<tbody style="line-height: 25px; font-size: 15px;">
    {% for user in obj2 %}
    <tr>
        <td scope="col" style="width: 50px; ">{{ user.id }}</td>
        <td scope="col" style="width: 480px; ">{{ user.institution }}</td>
        <td scope="col" style="width: 170px;">{{ user.city }}</td>
        <td scope="col" style="width: 230px;">{{ user.province }}</td>

    </tr>
  {% endfor %}
</tbody>
```

Furthermore it has a search filter. Selecting your province and city, it will filter out according to the ones you chose. Here's and example from the webpage:

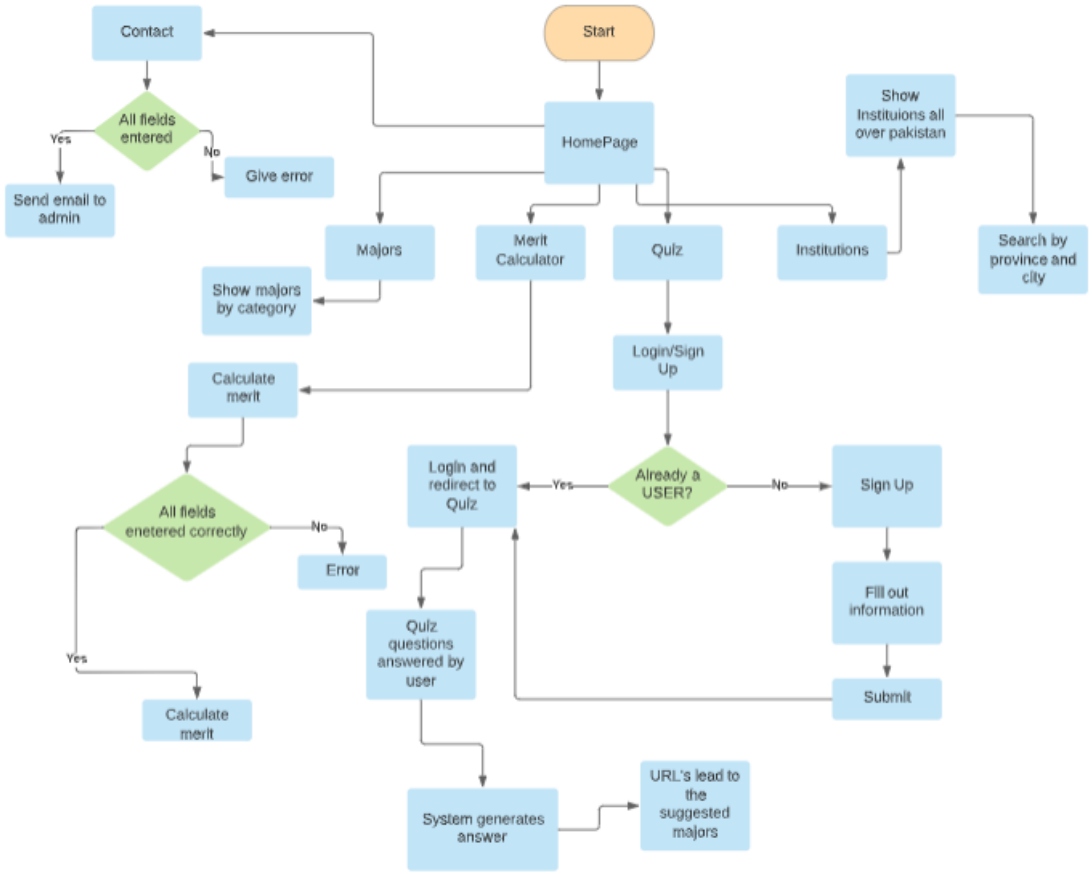
Explore Institutions

Refine your search by entering your city and province in the search filter below for more specific results.

PROVINCE Balochistan CITY Quetta

#	INSTITUTION	CITY	PROVINCE
33	Al- Hamd Islamic University	Quetta	Balochistan
92	Balochistan Agriculture College	Quetta	Balochistan
95	Balochistan University Of It & Management Sciences	Quetta	Balochistan
143	College Of Design [qta]	Quetta	Balochistan
251	General Musa Government Degree College	Quetta	Balochistan
293	Government College [quet]	Quetta	Balochistan
478	Government Girls College	Quetta	Balochistan
490	Government Girls Degree College[quarry Road]	Quetta	Balochistan
601	Government Science College	Quetta	Balochistan
706	Iqra University [qta]	Quetta	Balochistan
895	National Law College	Quetta	Balochistan

III. Project flow diagram



IV. Project repository link

Here is our project repository link

`https://gitlab.com/group12610/dsatermproj.git`

V. Division of tasks

Tasks between members were divided efficiently :

UI screens:

Ushna	Zainab
Login/SignUp	Merit
Home Page	Institutions
Quiz	Majors

Ushna was responsible for the backend of Quiz, Merit and Contact form. **Zainab** was responsible for the live deployment, the search engine, django documentation, and databases for the majors and institutions page.

VI. Finalized Project methodology

- **Analysis/Requirements**

  This web application was made for the sole purpose of guiding students and making it easier for them to find a career and a university. A student/teacher/parent or any one for that matter can easily browse through the various majors and institutions that offer that major. They can calculator their aggregate for different universities in Pakistan. This can help in narrowing down the options.

- **Design**

  The design stage helps in specifying hardware, software requirements and also helps in defining overall system architecture.

  `The software and hardware requirements:`

  – Operating system: Windows 8 or later

  – Processor: Intel Pentium 4 or later

  – Hard Drive: Minimum 32 GB; Recommended 64 GB or more.

  – Sublime

– Dreamweaver

– Python

The user interface was initially created using justinmind and Adobe XD. It was then later made on HTML and CSS.

- **Coding**

  After the completion of analysis and designing phase, the next phase is to develop a software system. We start by writing the code on any programming language such as Java, python, c++ etc. For our project we did our front-end using HTML and CSS, and the backend was done on JAVASCRIPT.

- **Testing**

  We then start checking the functionality of an application according to the requirements.

- **Installation**

  The application is then installed into the a particular environment

- **Maintenance**

  We will continue this phase until the project comes to an end.

VII. Deployment details

HEROKU and git was used for the deployment. The application was deployed by the the name fieldfinderapp.

VIII. Results and Evaluations

The application is working smoothly. It is compatible with most browsers and devices and is fully responsive.

IX. Challenges and Future Work

We ran into several problems while developing this application, mainly with the formulation of the quiz, getting the search filter to work properly, making sure the pages are all resposinve and finally, deploying it. There is clearly future work to be done by addressing the limitations of the application. Field Finder will introduce new features like:

Users will be able to apply to universities through it

A career page will be introduced

Post Graduate and Scholarships information will be added

X. References

Here is our reference list

https://www.eduvision.edu.pk/programs.php

w3schools.com/html/html_responsive.asp

https://en.wikipedia.org/wiki/List_of_universities_in_Pakistan

https://pietschsoft.com/post/2015/09/05/

javascript-basics-how-to-create-a-dictionary-with-keyvalue-pairs

https:

//gist.github.com/webbower/8d19b714ded3ec53d1d7ed32b79fdbac