Refactory Developer Challenge
Build a product: **ProfileMe.**

BUILD A PRODUCT: ProfileME.

Project Overview
Profile me Is an automated and computerized web-based system that aids Refactory staff input details of their students, then it prepares, generate and display a well arranged profile of that student.

Project Timelines
Total Duration: 3 Hours (2pm - 5pm)
Final Due Date : Monday 9th September, 2019.

Required Features
1. User (Staff) can enter student details.
2. User can submit student details.
3. User can cancel a data entry session.
4. User can view student profile only after entering student details.
5. User can close the Profile view page and return to the data entry page (form)
6. The application (system) should render and display the student details entry form, Profile view page, or any of those optional but relevant. Each should only be rendered when needed or requested for.
7. The application (system) should validate fields or data (student details) before being submitted.
8. The application (system) should clear and or reset all fields on cancelling a data entry session.
9. The application (system) should inform the User of a successful submission of the form or data and there and then provide two options, the first one (**view Profile)** for viewing the profile of the student basing on the submitted data, and another option (**Back**) that leads back to data entry page / form for a new data entry session.
10. The application (system) should allow exit of Profile Page

Optional Features (Not Mandatory)
1. User can Printout the Student Profile view page
2. User can login and logout of the system
3. System can store and retrieve data about Students in the database (**SQL** or **NoSQL**)

Note:
1. The features listed in the "**Required Features**" section are a must and hence have to be part of the system.
2. The features listed in the "**Optional Features (Not Mandatory)**" section are not a must be and hence you may only make them part of the system if you find the dedicated time enough for you, and it is highly advisable to attempt them last if you have to.

3. Validation of a particular Field can either be on change of that field or on submit of the data entry form or on any other event applicable to the field.

Preparation Guidelines
These are the steps you ought to take to get ready to start building the project

Steps
1. Check for whether you have git installed on the computer you are to use.

2. Login to your github account (Create one if you don't have).

3. Follow this link https://github.com/tech-refactory/Refactory-Catalyst002-Final-Assessment and **fork** the repository named Refactory-Catalyst002-Final-Assessment.

4. Now **clone** a forked copy of the above mentioned repository to any desired location (directory) of the computer like (desktop).

5. Open that cloned copy on your computer and then within it create an empty folder and name it with you First and Last name (or Sur and Given name).

6. Within that folder (the one with your name) is where you should have your project done. Ensure to create and use the **gitignore** file to ignore or the un-necessary files and folders with in your project so you don't make them part of your commits and pushes.

7. Now Develop your project as you add, commit, and push to update your forked repository (Don't Push to or try to update upstream (the central repository)) during development. You are expected to make only one pull request and that is at the end of your assessment. All pull requests will be merged once at the end of the assessment by a staff member (Administrator).

Challenge 1 – Student Data entry form Build

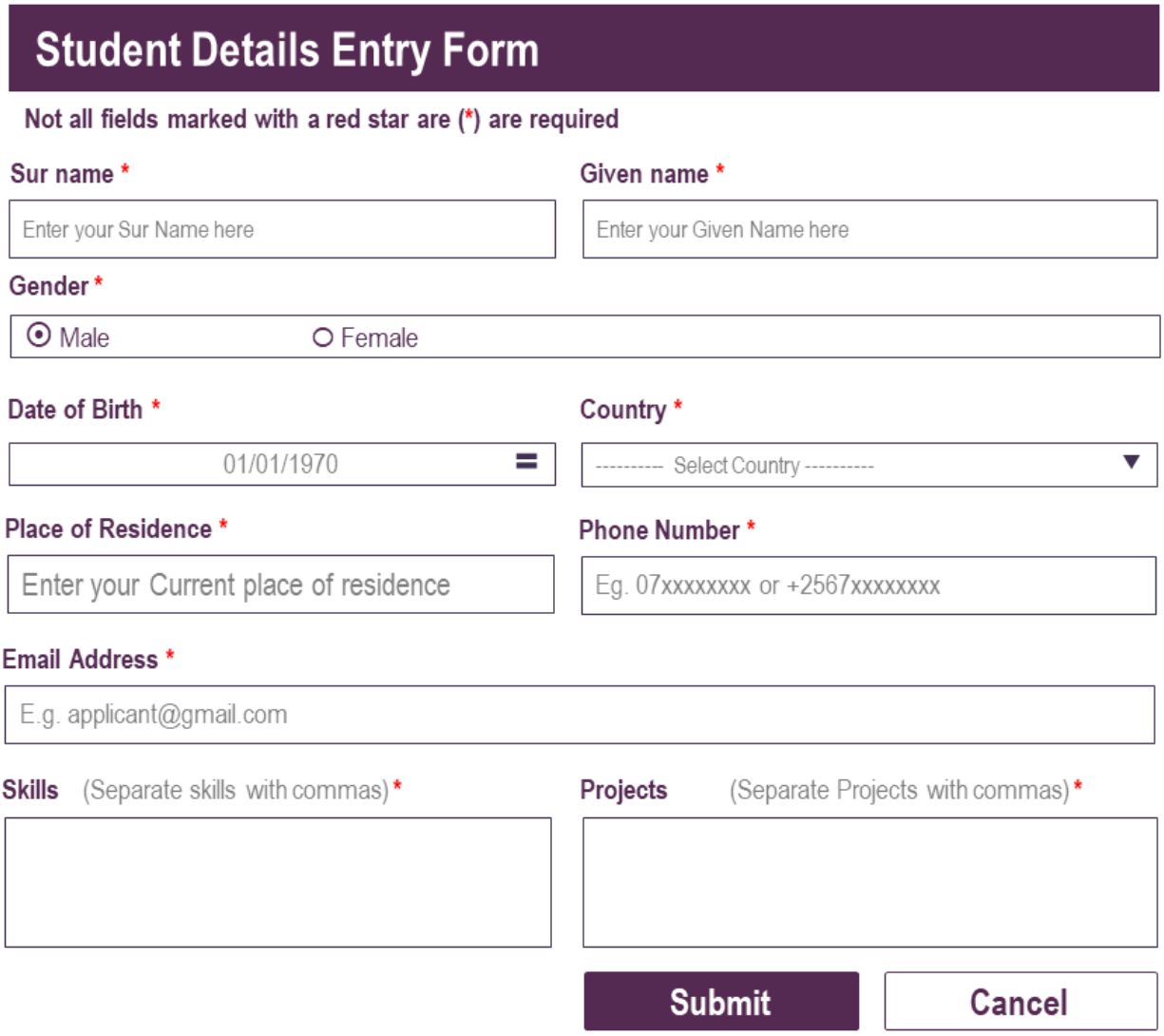
Challenge Summary
You are expected to create a front-end UI with HTML, CSS and JavaScript as the core technologies but you can optionally use frameworks and libraries (but only those hinted on in this guide) like **bootstrap, Jquery and VueJS**. This UI should be functional with fields ready for data input by the user.

Note that this challenge requires that you implement your front-end logic using standard language capabilities (e.g. HTML5, CSS3, and (ES5 or ES6 for JavaScript)).

Guidelines

1. Create a data entry form which is to capture student details:

The Entry from is expectedly to look like as below



If you have to apply HTML standardization and validation, it must be at this level. i.e. ensuring to set attributes for the respective input fields to make them best suit thhe expected data.

Tools

Below is a list of tools and technologies from which a student can choose what to use to complete the first challenge

- HTML5
- CSS3

- BOOTSTRAP
- VueJS
- NodeJS

Challenge 2 – Student Data entry form Validation

Challenge Summary

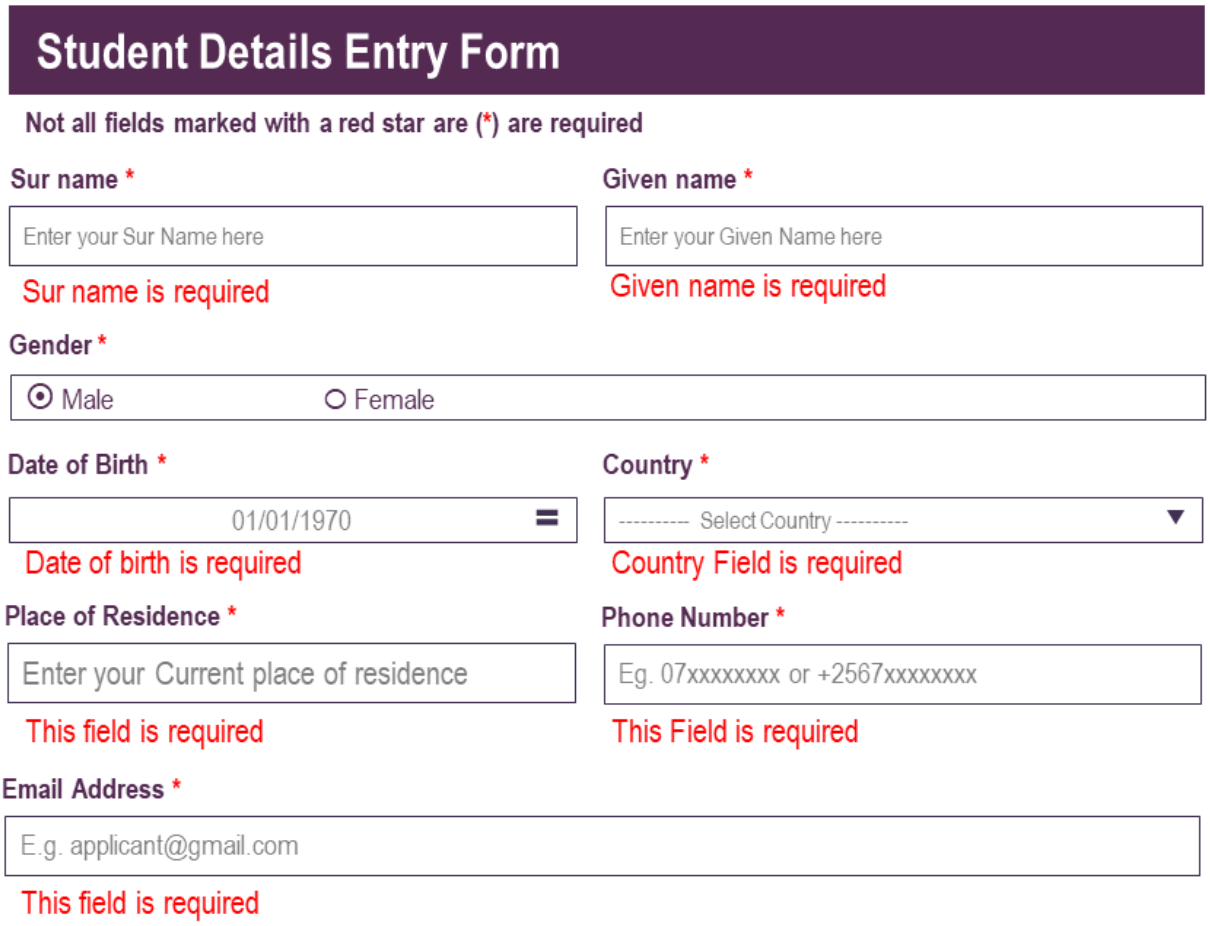
You are expected to validate the form created in challenge 1 with JavaScript as the core technologies but you can optionally use technologies, frameworks and libraries (but only those hinted on in this guide) like **HTML, JavaScript, Jquery and VueJS**. This is to make the UI functional such that on submit, data must be valid to make a successfull submission.

Note that this challenge requires that you implement your front-end logic using standard language capabilities (e.g. HTML5, CSS3, and (ES5 or ES6 for JavaScript)).

Guidelines

1. Validate each required field:

Each error label should have font color as red and should be below the field corresponding to the validation at a particular instance. See image below.



Apply Javascript validation skills to achieve such as above.

If all the fields are (or the data in each) is valid, then the form must be ready for submission.

Tools

Below is a list of tools and technologies from which a student can choose what to use to complete the first challenge

- HTML5
- CSS3
- BOOTSTRAP
- VueJS
- NodeJS

Challenge 3 – Form Submission and Profile Page rendering.

Challenge Summary

You are expected to comeup with the code to handle form submission with JavaScript as the core technologies but you can optionally use technologies, frameworks and libraries (but only those hinted on in this guide) like **Jquery and VueJS**. This is to ensure that the form developed in challenge one and validated as in challenge 2 can be successfully submitted and or the user is informed on success in submission and otherwise.

Guidelines

1. In case of any failure in submission of a well validated form, the system should display an alert message that says: "*Problem encountered while submitting this form. Please try Re-submitting "* and then the form should stay rendered.

2. In case of success in submission of a well validated form, the system should display an alert message that says: "*Data has been successfully submitted.*" and then the system should display the student profile page with the submitted.

3. You can use any of the options below to help store your submitted data.

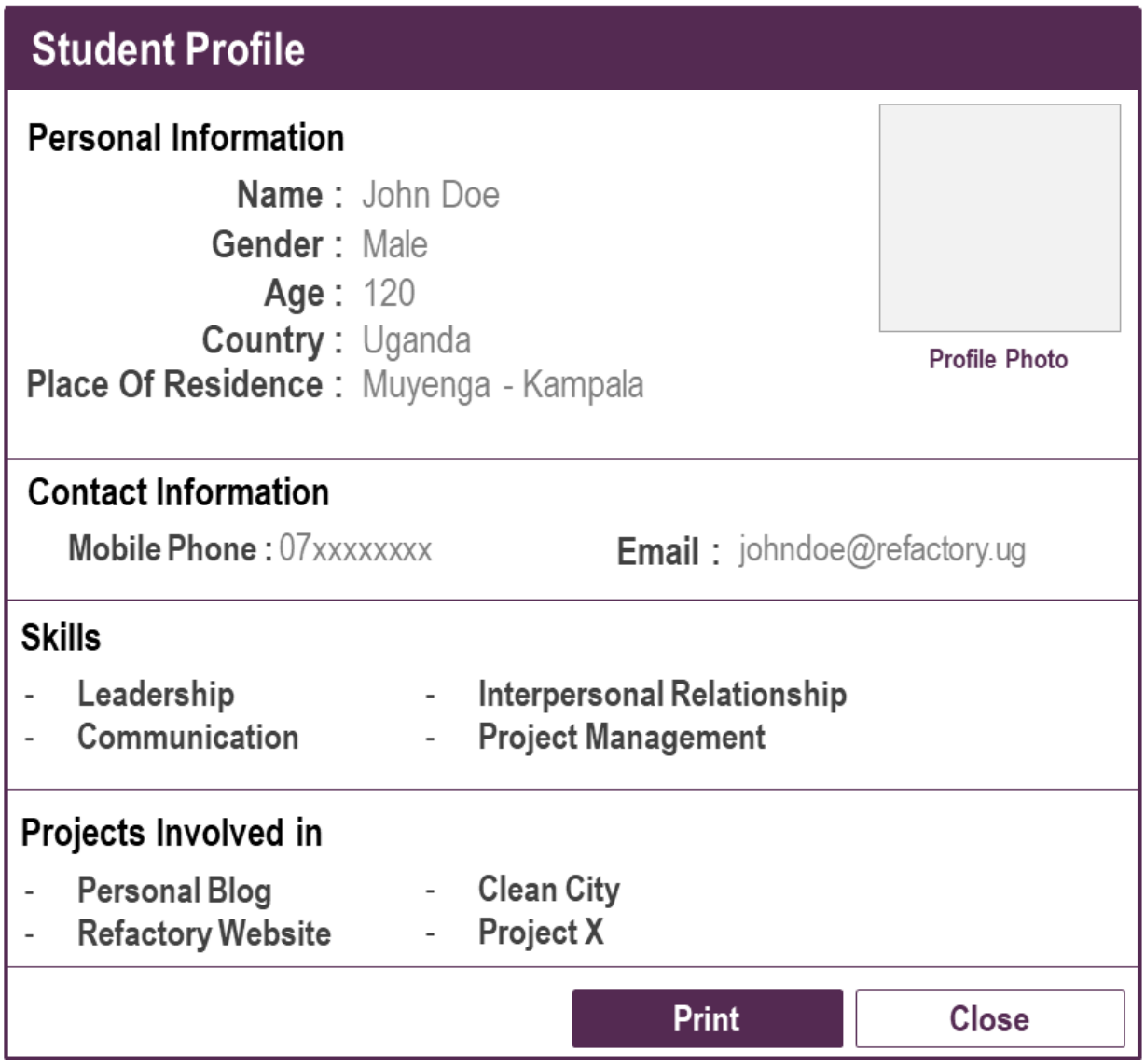
    i) You can simply use a javascript literal object named **Student.** With properties as in the form in challenge 1 and 2. Then use the object to serve data to the Profile page.

    ii) You can store your data in a JSON file name **Student** from which you can read and serve data to the Profile page.

    iii) You can create a Database named **refactory** and then create a table called **student**(For SQL), or a collection called **student**(for NoSQL)store data into it which you can then retrieve and serve the profile page. The database can be SQL(MySQL) or NoSQL(MongoDB).

    Note: use the javascript standard alert() function to implement all the alerts.

Below is how the profile page should look like.



Design and implement the profile photo section but no image is required for it.

Tools

Below is a list of tools and technologies from which a student can choose what to use to complete the first challenge

- HTML5
- CSS3
- BOOTSTRAP
- VueJS
- NodeJS
- MySQL

- MongoDB