November 2022

Refactory

Catalyst 012

Final Technical Assessment

REFACTORY CATALYST12 FINAL TECHNICAL ASSESSMENT

Introduction:

Refactory Uganda is a Program aimed at skilling Software Developers and render them ready for the job market. Literally, Refactory registers software development enthusiasts, those that are completely green about software development (The green horns), or those looking at upskilling. After registration, the registered students are then taken through a series of skilling programs till when the students are turned into fully fledged software developers. Because Refactory Has been doing this for years, and has proved to be good at it, the number of applicants for the program drastically rise for every new Cohort and the currently deployed Manual student registration process is seemingly becoming hectic and inefficient.

Challenge:

In attempt to address the challenges identified in the current student registration process at Refactory, the Refactory Technical Department Proposed an electronic Information system to Digitize and Automate the processes. The proposal, was approved, the system was documented, planned, and designed as well, and now all efforts are directed towards implementation of the system. You are one of the selected qualified and technically competent full stack Software developers and You have been assigned the “Student Registration form” module. Prove your competence by honoring and fully addressing the challenge presented to you.

Proposed Module development time:

Total Duration: 03 Hours

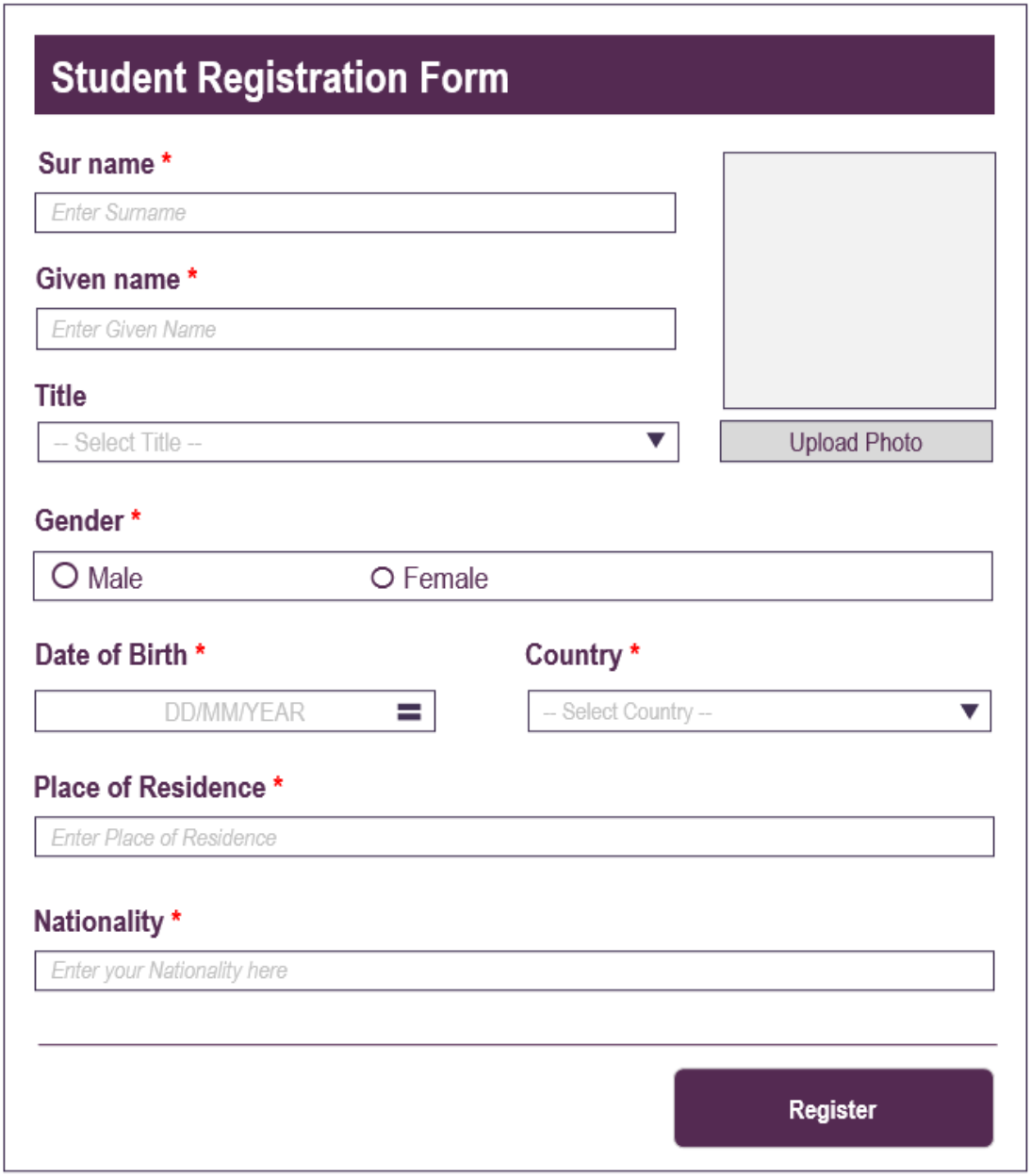
Requirements [What to be Implemented]:

- System should render the registration form upon request.
- User can enter details of Student into the registration form.
- User can submit the registration form (with Student details)
- Upon form submission, the system should validate all fields.
- Upon form validation, if any field is invalid, the system should Inform the user about the invalidity and terminate the submission process.
- After form validation, the system should submit a form only if all fields are valid.

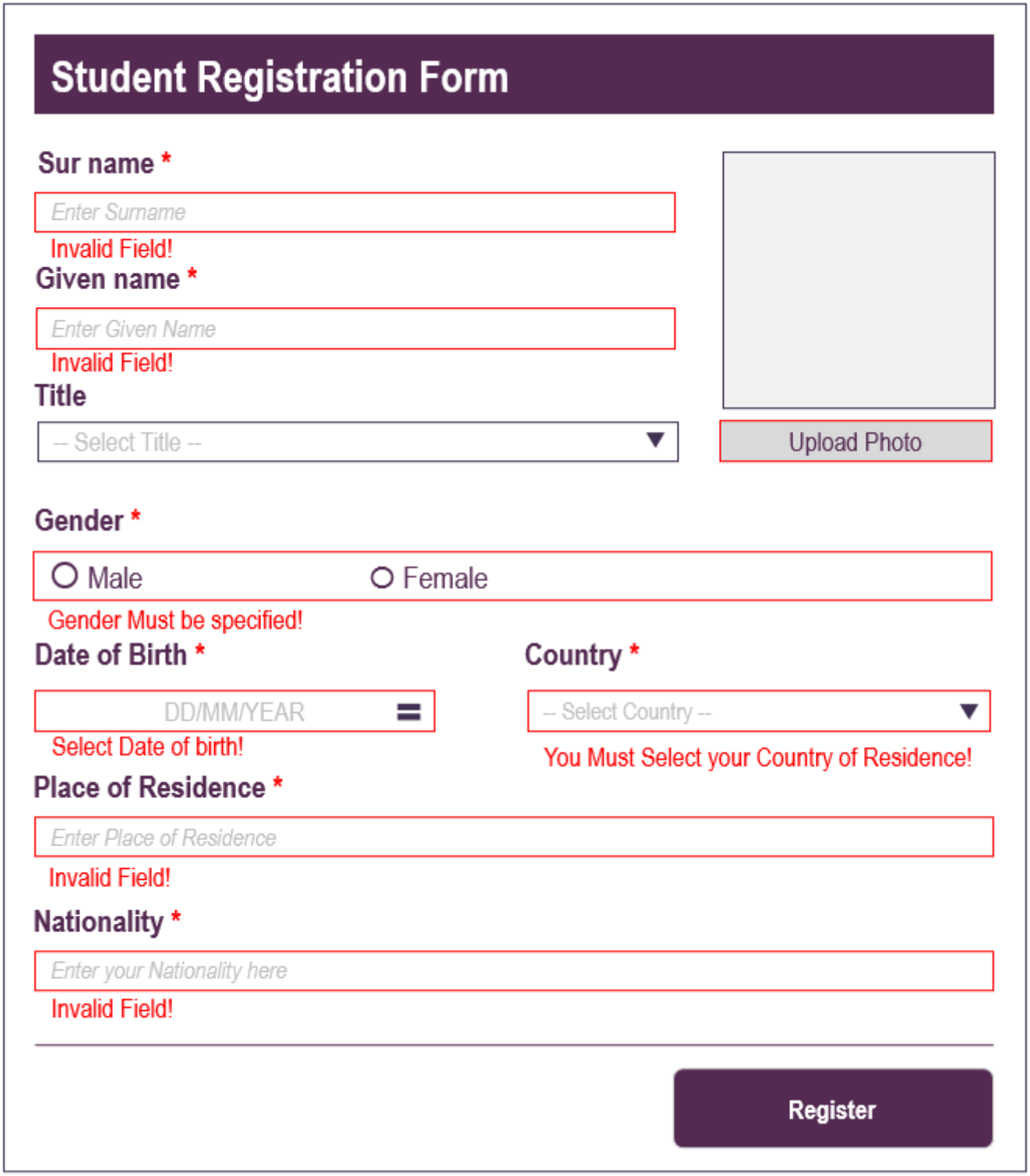
- After form submission, the system's Resource server should receive the form data and store it in the database.
- After a successful form submission, Validation and data storage, the system should redirect to the same form, reset the form fields and notify the user with a success message.
- The success message should disappear upon close or upon focus of any of the form elements.

Proposed User Interfaces according to the form designs:

According to the designs of the proposed system, after requesting for the Registration form from the server, the form is expected to look as in the Figure below after it's initial rendering.



Below is the proposed look for each of the invalid fields after form validation.



Below is the proposed look for the Registration form immediately after a successful Student registration.

Student Registration Form

Registered Successfully

Sur name *

Doe

Given name *

John

Title

Dr.

Upload Photo

Gender *

(•) Male ( ) Female

Date of Birth *

01/01/2019

Country *

Uganda

Place of Residence *

Muyenga - Kampala

Nationality *

Ugandan

Form Validation Guidelines:

Validation of the Form Fields Should be done basing on the following guidelines:

1. Surname Field:

It should be between 1 to 16 alpha-bet characters (limits exclusive)

2. Given Name Field:

It should be between 1 to 16 alpha-bet characters (limits exclusive)

3. Date of Birth Field:

Registration can only happen to and by a Student who is at least 16 years old and at most 150 years old.

4. Place of Residence Field:

It should be between 1 to 20 alpha-bet characters (limits exclusive)

5. Title:

This field is optional to the system users however, These are the possible options from which a user can select: [Mr., Mrs., Miss, Dr. Prof., Eng.]

6. Nationality Field:

It should be between 5 to 20 alpha-bet characters (limits exclusive)

7. Gender Field:

Student can be Male or Female but neither none of the two nor both. though set Male as default

8. Country field:

This field represents a category of a patient being registered, which is to be selected from the availed list of categories as below.

-- Select Country – should be used as a place holder (Automatically set value as default value) and hence should not be considered as a valid Country input value.

A valid Country input value can only be one and only one of the following options.

- Uganda
- Kenya
- Tanzania
- Rwanda
- Burundi

Programming Languages, Libraries, Frameworks and tools to be used:

1. HTML
2. CSS
    - Bootstrap (optional)

3. JavaScript
4. Nodejs
    - Vue.js (optional)
    - Express
    - Body-Parser
    - Pug (optional)
    - Mongoose
    - Passport.js (optional)
    - bcrypt (optional)
    - Multer (Optional)

5. Git
    - Git
    - GitHub
    - GitHub Desktop

6. MongoDB
    - MongoDB Compass
    - Mongo Atlas

Text Editors and IDEs:

- Visual Studio Code (VScode).

Preparation Guidelines:

Note: Assessment Starts right from here

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

Step 1: Check for whether you have Git installed on the computer you are to use.

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

Step 3: Follow this link https://github.com/tech-refactory/Refactory-Catalyst012-Final-Technical-Assessment and fork the repository named **Refactory-Catalyst012-Final-Technical-Assessment**

Step 4: Now clone a fork (copy) of the above-mentioned repository to any desired location (directory) on your computer (like on the desktop).

Step 5: Open the **Cloned repository** in a text editor and a terminal or Console input and output window (Most preferably VScode with Git Bash as the integrated terminal) Otherwise if you are to use a text editor with no integrated Console Terminal, Open the Cloned repository in any terminal availed by your Operating system, i.e. **CMD**, **POWERSHELL LINUX TERMINAL, etc**.

> **Note:** Every time you are to stage changes, commit changes or push changes to a remote repository, ensure that the path is pointing to this cloned repository.
>
> **Note:** Endeavor not to Initialize any repository within this repository.
>
> **Note:** Ensure to have the gitignore file (if not create one) right at the start of this repository. Use it to ignore all the un-necessary files and Directories with in this repository so you don't make them part of your commits and pushes.

Step 6: 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).

> **Note:** You may receive Instructions from your Invigilator or Instructor in a refined way from what is in this step (Step 6)

Step 7: Within that folder (the one with your name) is where you should have your project done. Now start and work on your project as you add, commit, and push to update your remote repository.

> **Note:** The repository you push changes to should be (a fork) the one you attained after forking the central repository (as in step3). Don't Push to (or try to update) the central repository during development.
>
> **Note:** 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 the Administrator of the central repository.

Success wishes from

The Refactory Technical Department

On behalf of Refactory