



CSE-3200: System Development Project

OMR Sheet Evaluator Using Image Processing

By

Name: Shadman Rabby

Roll: 1807085

Name: Arnob Sarker

Roll: 1807088

Supervisor:

Dola Das

Assistant Professor

Department of Computer Science and Engineering

Khulna University of Engineering & Technology

Department of Computer Science and Engineering

Khulna University of Engineering & Technology

Approval

This Project Report has been submitted for examination with the approval of our supervisor.

Dola Das

Assistant Professor

Department of Computer Science and Engineering

Khulna University of Engineering & Technology

Acknowledgement

We are thankful to our supervisor, **Dola Das** to give us proper guidelines to complete our project successfully. We have completed our “**CSE 3200 System Development Project**” titled “**OMR Sheet Evaluator using image processing**” with help of our supervisor.

Shadman Rabby

1807085

&

Arnob Sarker

1807088

Abstract

OMR sheets have been widely used across the globe for grading purposes. It is a process of evaluating human-marked data from documents such as surveys and tests. This technology acts as a guide or helping tool for processing large amount of data, mainly questionnaires and multiple-choice questions. Almost all the educational institutions in our country use OMR technology. The exam taker has to feel the bubble or the box as an appropriate answer to the question. The following proposed system helps to scan the input sheets and store the result in the database, so the user can see the saved scores. It is done by using various image processing techniques such as Gaussian filtering, BGR to Gray, edge detection, thresholding, compliment of the image. The answers marked for each of the questions help evaluate the sheet and display the total marks. This system eliminates the installation of heavy machinery, and expensive scanners in return saving time and cost. The proposed work consists of an ordinary smartphone camera and a computer for computation purposes.

Contents

Approval	i
Acknowledgement	ii
Abstract	iii
Chapter 1: Introduction	
1.1 Introduction	1
1.2 Background	1
1.3 Features	1
Chapter 2: Methodology	
2.1 Technology Used	2
2.2 Methodology	2
2.2.1 Processing the input image	2
2.2.2 Interfacing between php and python file	8
2.2.3 Website Development	8
2.2.4 Database	9
Chapter 3: Result	
3.1 Result	9
3.2 Graphical User Interface (GUI).....	9
3.3 Performance Evaluation	13
Chapter 4: Conclusion	
4.1 Conclusion and future work	14
References	14

Chapter 1: Introduction

1.1 Introduction

Optical Mark Recognition (OMR) is to scan a paper to detect the presence of the mark in a specified location on the sheet. Today OMR is harnessed as the input for an data entry system. Previously it had forms of paper tape and punch cards. These had real holes punched in them instead of the marked circles. OMR sheets have replaced by the subjective answer sheets and are widely used as a evaluation or grading tools for the exams conducted by the schools, colleges and many other institution. It has made the tedious job of grading the exam taker easier as the examiner does not have to read through the paragraphs of the answers written by the exam taker. For example, competitive exams in Bangladesh stick to the OMR sheets for the calculation of marks. OMR is generally used as direct input for data of censuses and surveys and is perfect for discrete data handling, whose values are limited set. In the field of education widely uses OMR technique can process objective questions in exams,. Earlier OCR used a optical scanner to scan the paper as bitmap image and the software was used to realize the texts and the images. These OCRs matched these input images with the stored bitmap images which resulted in inaccuracy due to its hit and miss pattern recognition. Here in our system we have used different kind of image processing techniques to detect the rolls and answers accurately.

1.2 Background

The purpose of this website is to provide users the facility to evaluate multiple hand filled OMR without using any heavy machine or device , just need a laptop or computer where the image of the OMR sheets are stored. Not all the institutions or teachers own a OMR machine ,so question setter find it difficult to check the right answers for every students OMR. On those case our system can be a good solution for them as it is available for everyone and easy to use.

1.3 Features

- Customizable actual answer OMR sheet.
- User can also upload the image of the hand filled right answer sheet.
- Batch mode facility (Multiple OMR can be uploaded at a time).
- User can input the number of questions and number of options per question.
- Display the score
- Show the archive (previous results)

Chapter 2: Methodology

2.1 Technology Used

Software:

- Microsoft Visual Code (VS code)
- XAMPP v3.3.0
- Web Browser (Google chrome)
- Windows 10

Languages:

- HTML
- CSS
- Java Script
- PHP
- SQL
- Python

2.2 Methodology

There can be two types of input image.

- Actual answer sheet (Teachers copy)
- Students answer sheet

All the implementation steps are discussed below:

2.2.1 Processing the input image

I. Resizing the image:

First of all, the input OMR is resized to fixed value of height and width. We set this size to 1000 X 800. This is done because image can be of any size.

II. Gray Scale conversion

Grayscale is the process of converting an image from other color spaces e.g., RGB, CMYK, HSV, etc. to shades of gray. It varies between complete black and complete white.

Why do we use grayscale:

Dimension reduction: For example, In RGB images there are three color channels and three dimensions while grayscale images are single-dimensional.

Reduces model complexity: Consider training neural articles on RGB images of 10x10x3 pixels. The input layer will have 300 input nodes. On the other hand, the same neural network will need only 100 input nodes for grayscale images.

For other algorithms to work: Many algorithms are customized to work only on grayscale images e.g. Canny edge detection function pre-implemented in the Opencv library works on Grayscale images only.

code:

```
import cv2

image = cv2.imread('images.jpg')

cv2.imshow('Original', image)

grayImage = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

cv2.imshow('Grayscale', grayimage)
```

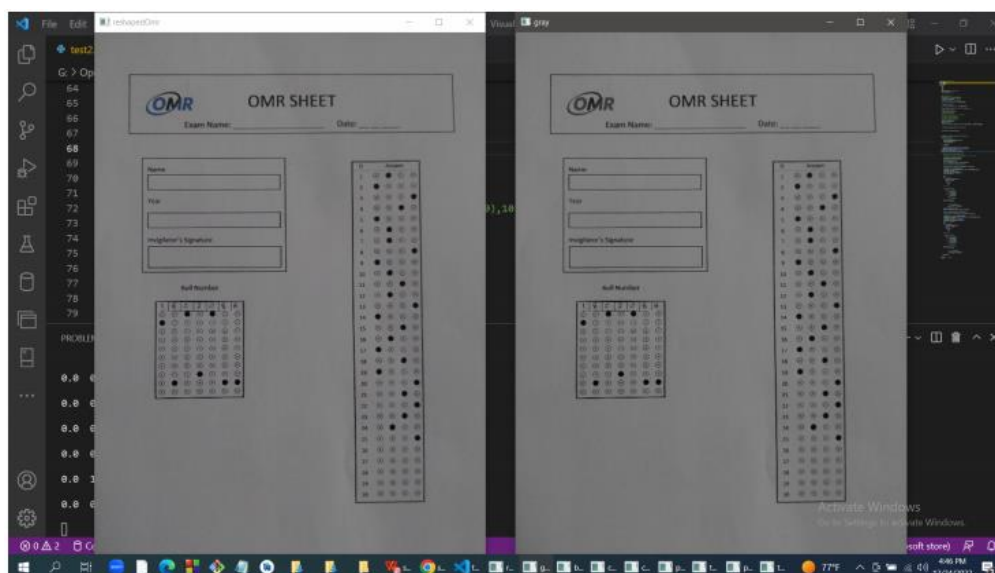


Figure 1: Input image to Gray Scale

III. Gaussian Blur

Image blurring is achieved by convolving the image with a lowpass filter kernel. It is useful for removing noise. It actually removes high frequency content (e.g.: noise, edges) from the image. So, edges are blurred a little bit in this operation. OpenCV provides four main types of blurring techniques. We will use the Gaussian Blur technique.

code:

```
blur = cv.GaussianBlur(img,(5,5),0)
```

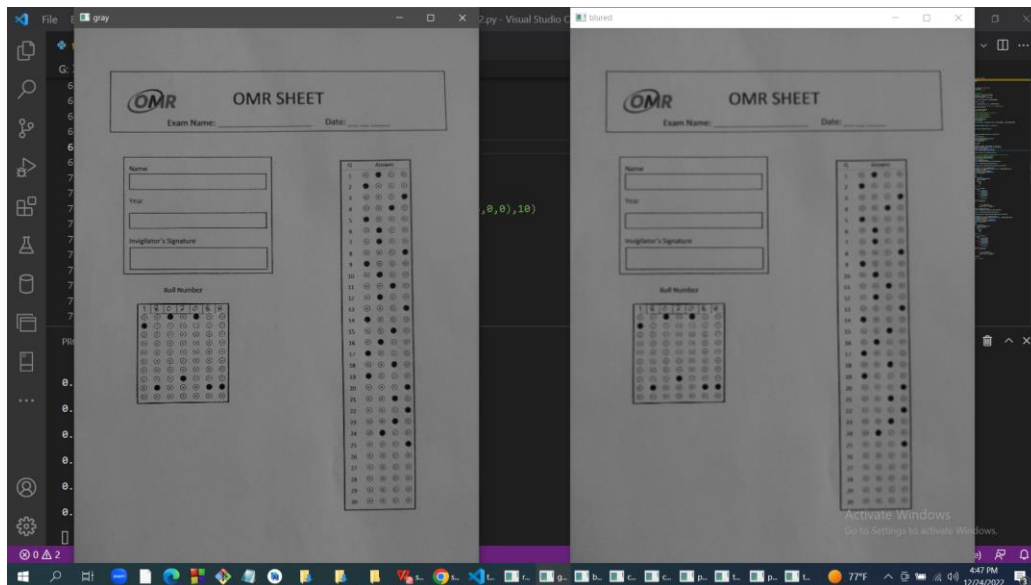


Figure 2: Gray to Blurred

IV. Edge Detection:

Canny Edge filter in OpenCV. Canny() Function in OpenCV is used to detect the edges in an image.

Syntax: `cv2.Canny(image, TLower, TUpper, apertureSize, L2Gradient)`

Where:

Image: Input image to which Canny filter will be applied

TLower: Lower threshold value in Hysteresis Thresholding

TUpper: Upper threshold value in Hysteresis Thresholding

ApertureSize: Aperture size of the Sobel filter.

L2Gradient: Boolean parameter used for more precision in calculating Edge Gradient.

Canny Edge detection is an Algorithm consisting of 4 major steps:

- Reduce Noise using Gaussian Smoothing.
- Compute image gradient using Sobel filter.
- Apply Non-Max Suppression or NMS to just keep the local maxima
- Finally, apply Hysteresis thresholding which that 2 threshold values TUpper -and TLower which is used in the Canny() function.

code:

```
import cv2
img = cv2.imread("test.jpeg")
edge = cv2.Canny(img, tlower, tupper)
cv2.imshow('original', img)
cv2.imshow('edge', edge)
cv2.waitKey(0)
```

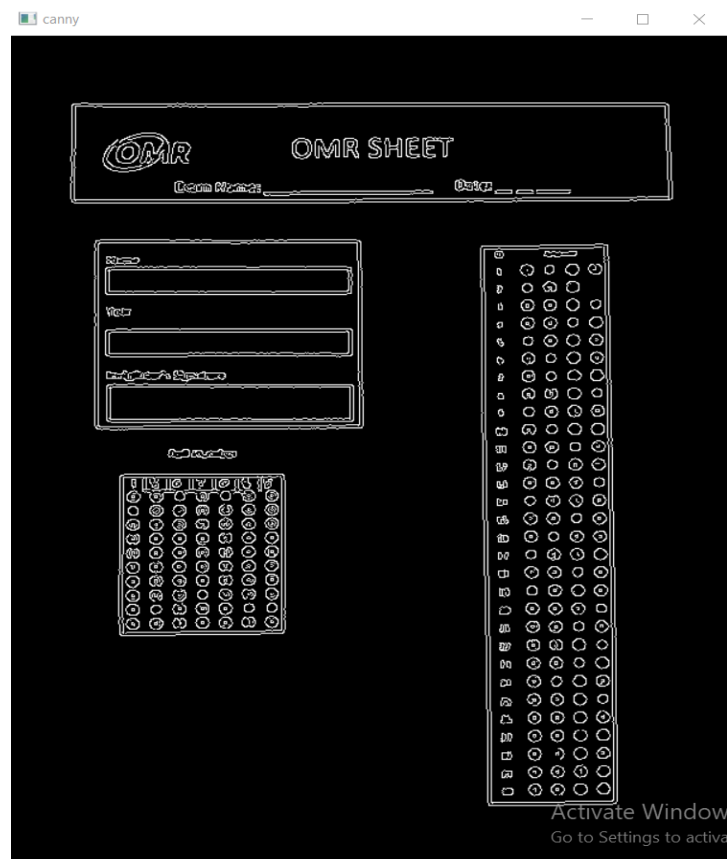


Figure 3: Edges detected

V. Separating the Rectangles from image

Before thresholding we need to the rectangles where the bubbles or circles are filled up. In our customize OMR sheet the largest rectangle consist the Questions and options part but the 4th rectangle has the circles to filled up for the student Roll No.to separate these images from the OMR sheet we use two functions of OpenCV:

```
matrix = cv.getPerspectiveTransform(point1,point2)
```

```
Imgwarpcolored = cv.warpPerspective(imgcpy2,matrix,(width,height))
```

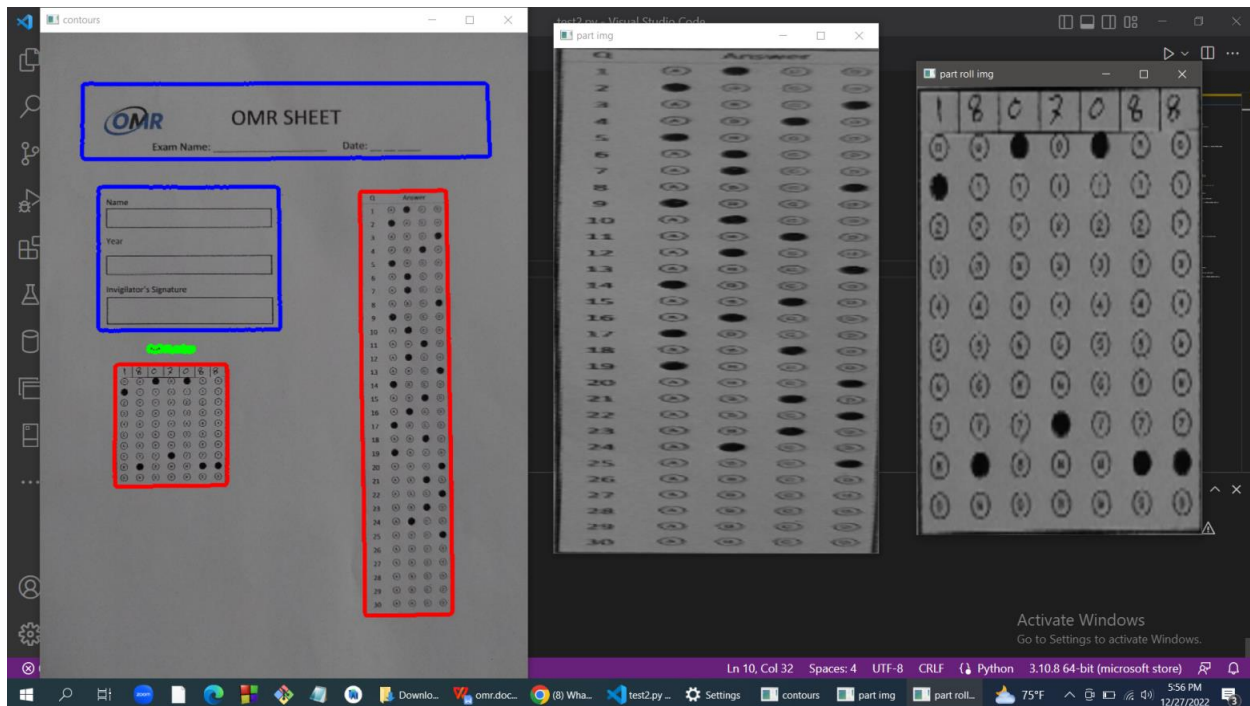


Figure 4: Separating the rectangular area where MCQ options and Roll are filled

VI. Binary Thresholding

For every pixel, the same threshold value is applied. If the pixel value is smaller than the threshold, it is set to 0, otherwise it is set to a maximum value. The function `cv.threshold` is used to apply the thresholding. The first argument is the source image, which should be a grayscale image. The second argument is the threshold value which is used to classify the pixel values.

The third argument is the maximum value which is assigned to pixel values exceeding the threshold. OpenCV provides different types of thresholding which is given by the fourth parameter of the function. Basic thresholding as described above is done by using the type `cv.THRESH_BINARY`. We used the `cv.THRESH_BINARY_INV` function from all the simple thresholding method.

Code:

```
cv.imshow('part img',imgwarpcolored)
```

```
imgthresh=cv.threshold(imgwarpcolored,75,255,cv.THRESHBINARYINV)
```

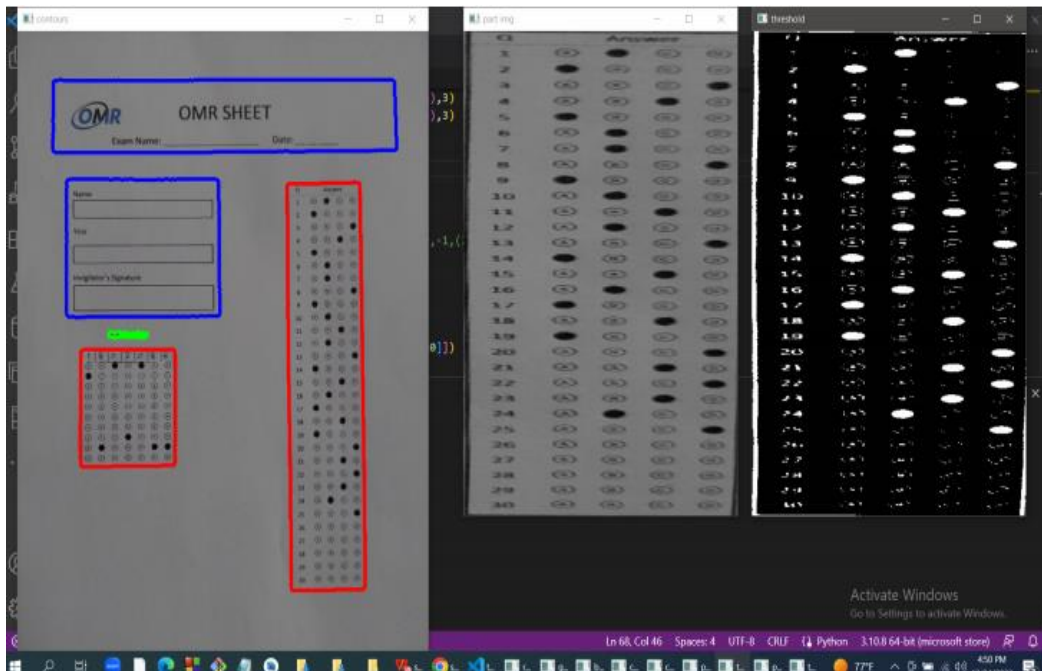


Figure 5: Thresholding the MCQ part image

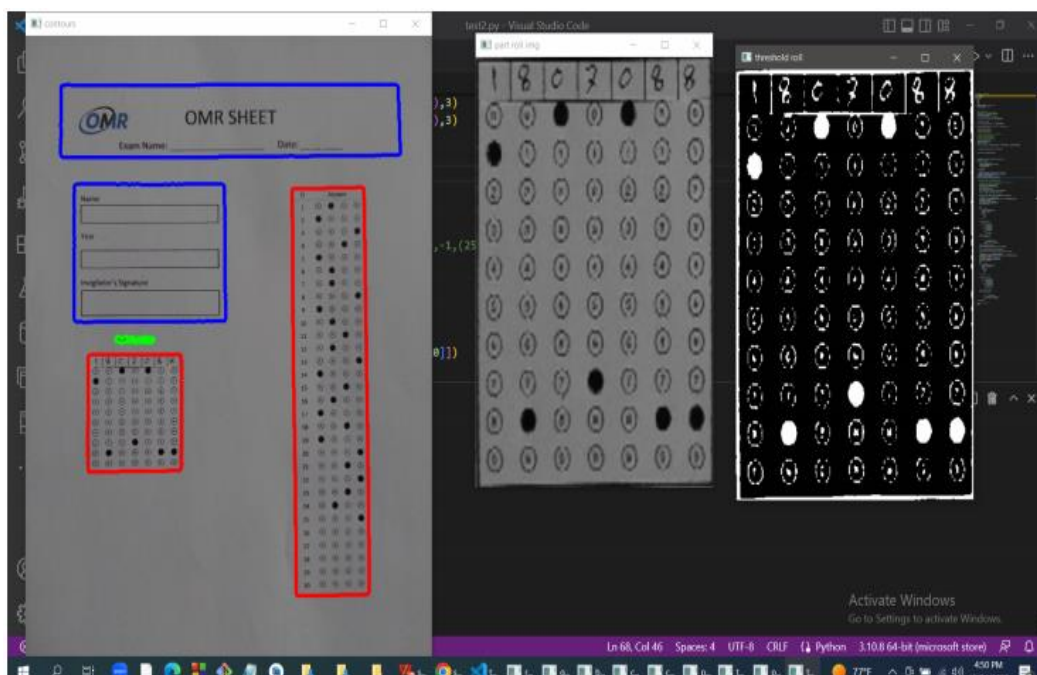


Figure 6: Thresholding the Roll No part image

2.2.2 Interfacing between PHP and Python file

We run the python code from php by shellexec() function and pass the image path as the argument. From python code we pass the roll number and the answers of the MCQs in JSON format. Then in the php code we decode the json format and work with those values. Now we check the result in PHP and store it in the database against the roll number.

2.2.3 Website Development

I. Frontend

The front end of a website is everything the user either sees or interacts with when they visit the website. It is responsible for the total look and feel of an online experience. Put simply, the front end is a combination of two different elements: the graphic design (the look) and the user interface (the feel). Each of these is created independently. Users mostly interact with front end. We designed our website using -

- HTML
- CSS
- BOOTSTRAP (Web framework)
- JavaScript

II. Backend

The backend, also called server-side, is the infrastructure that supports the front end and is made up of parts of a piece of software regular users can't see. The backend is basically a website's brain. The backend includes the server that provides data whenever requested, the database where that data is organized, and the application that delivers that information. To turn a new website into a dynamic web application (a website whose content may change depending on what is in its database and that can be changed by user input), you will need to add more backend components. This is different from a static website, where the content often remains the same and doesn't need a database. In the backend we had to handle everything that doesn't involve providing a user interface. This includes building libraries, and utilities.. To put it simply, We worked at backend to create code to ensure everything functions correctly in the frontend. We used PHP as our backend language. For database we used SQL server. To process our image Python is used.

2.2.4 Database

A database stores website content in a structure that makes it easy to retrieve, organize, edit, and save data. It runs on a remote computer called a server. There are many different databases that

are widely used, such as MySQL, SQL Server, PostgreSQL, and Oracle. For our project MySQL is used to store, retrieve and organize data. There are four tables in the database. Their physical schema is given below:

- exams (xmname,xmcode, id ,noofques, json)
- results (stdID, fullmarks, obtainedmarks, id, image,json)
- userinfo (ID, FirstName, LastName, Phone, Address, About, Organization,image)
- users (ID, name, email, password, created-at)

Chapter 3: Result

3.1 Result

i. Testcase1 (Our customize OMR)

For our customize OMR it can detect the edges accurately. Then it can separate the answer and Roll number part as new image from the total OMR image.

ii. Testcass2 (OMR of any Institutions)

It can detect the OMR which is not our customized one but it cannot give the correct result for that sometimes. When the edges are clear and bold then it can detect the rectangles otherwise it fails to separate the rectangles of rolls and answer circles (within the largest rectangle).

3.2 Graphical User Interface (GUI)

- **Homepage of the website**

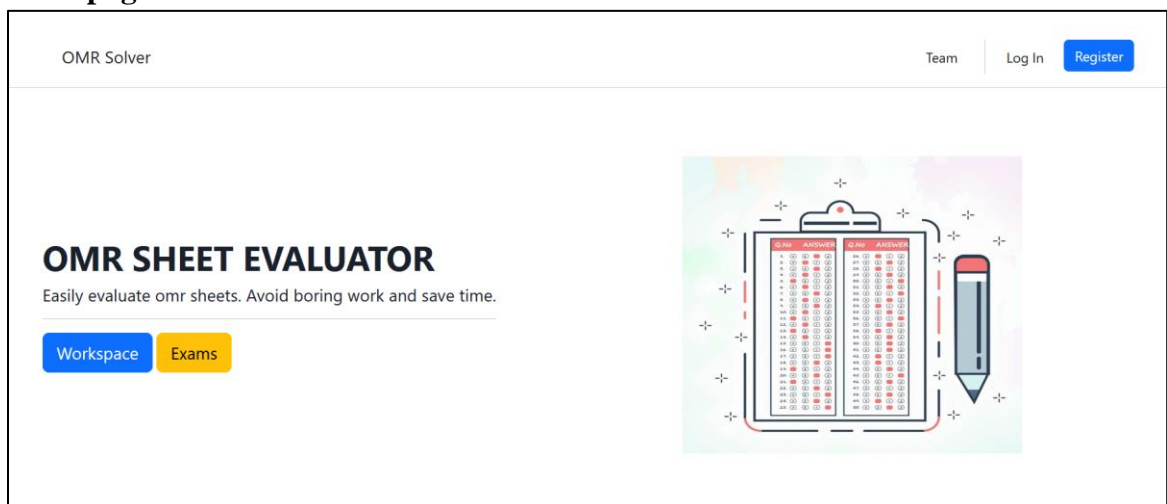
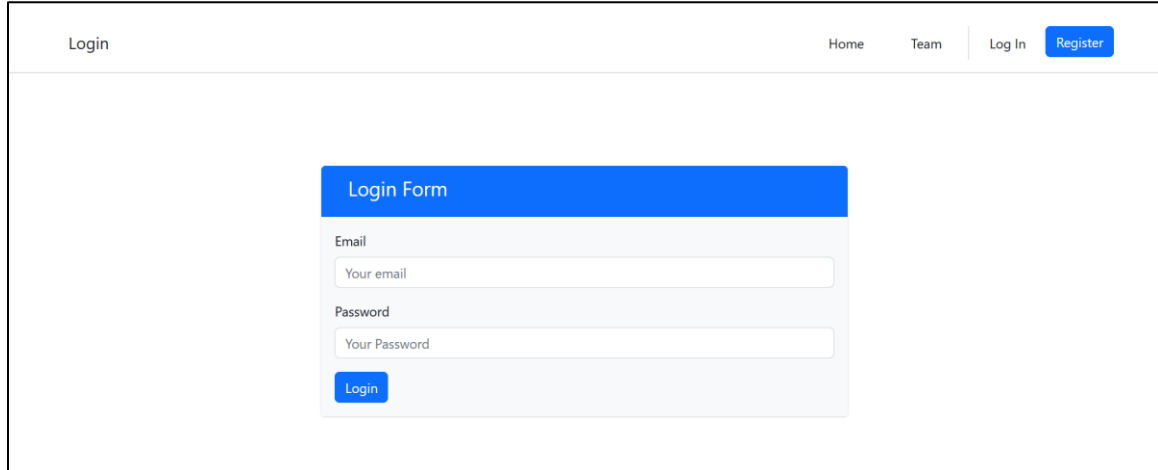


Figure 7: Homepage

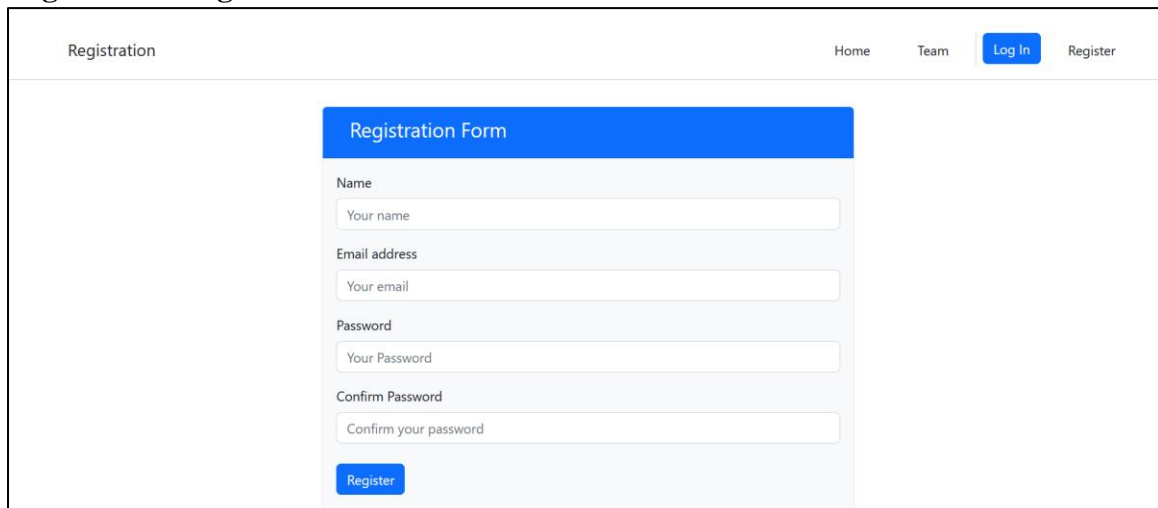
- **Login Page**



The screenshot shows a web application interface for the login page. At the top, there is a navigation bar with the text 'Login' on the left and links for 'Home', 'Team', 'Log In', and a blue 'Register' button on the right. The main content area features a 'Login Form' with a blue header. Below the header, there are two input fields: 'Email' with the placeholder text 'Your email' and 'Password' with the placeholder text 'Your Password'. A blue 'Login' button is positioned below the password field.

Figure 8: Login page

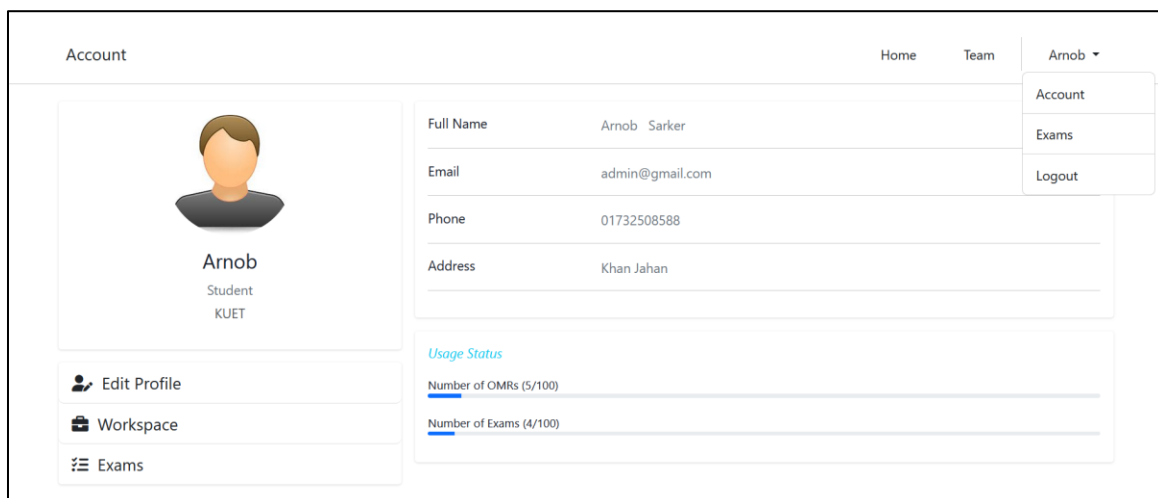
- **Registration Page**



The screenshot shows a web application interface for the registration page. At the top, there is a navigation bar with the text 'Registration' on the left and links for 'Home', 'Team', a blue 'Log In' button, and a 'Register' button on the right. The main content area features a 'Registration Form' with a blue header. Below the header, there are four input fields: 'Name' with the placeholder text 'Your name', 'Email address' with the placeholder text 'Your email', 'Password' with the placeholder text 'Your Password', and 'Confirm Password' with the placeholder text 'Confirm your password'. A blue 'Register' button is positioned below the confirm password field.

Figure 9: Registration page

- **User Account**



The screenshot shows a web application interface for the user account page. At the top, there is a navigation bar with the text 'Account' on the left and links for 'Home', 'Team', and a dropdown menu for 'Arnob'. The dropdown menu contains options for 'Account', 'Exams', and 'Logout'. The main content area is divided into two columns. The left column features a user profile card with a placeholder image, the name 'Arnob', and the role 'Student KUET'. Below the card are three buttons: 'Edit Profile', 'Workspace', and 'Exams'. The right column displays user information in a table format:

Full Name	Arnob Sarker
Email	admin@gmail.com
Phone	01732508588
Address	Khan Jahan

Below the table, there is a section titled 'Usage Status' with two progress bars:

- Number of OMRs (5/100)
- Number of Exams (4/100)

Figure 10: Account page

- **Edit Profile**

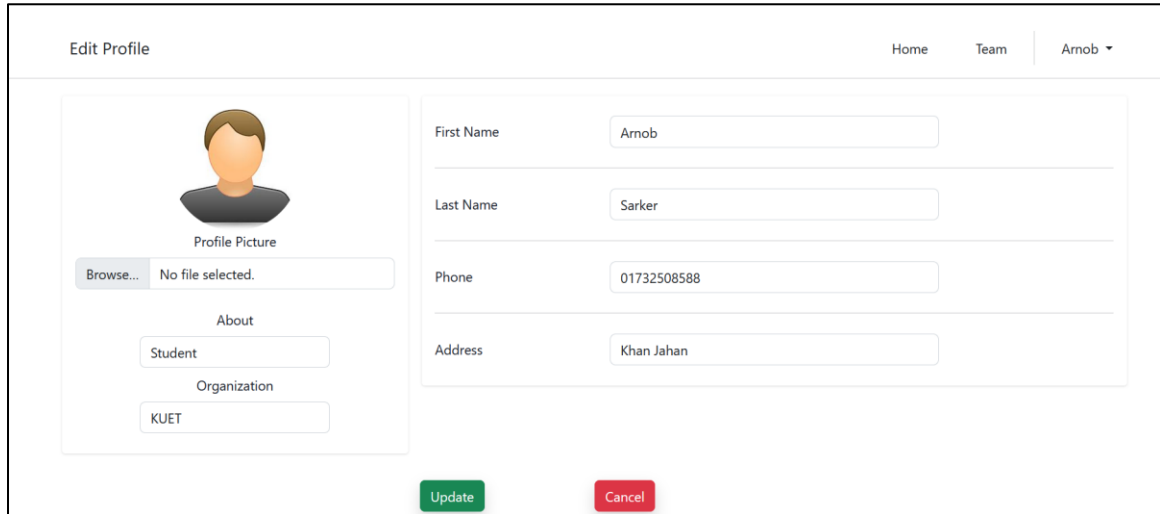


Figure 11: Edit profile page

- **Workspace**

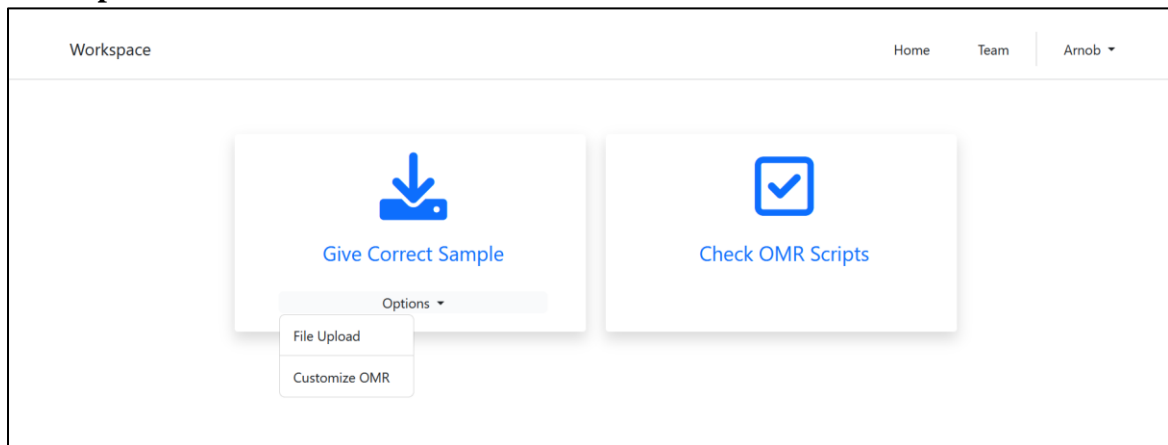


Figure 12: Workspace page

- **Give exam details and submit correct OMR**

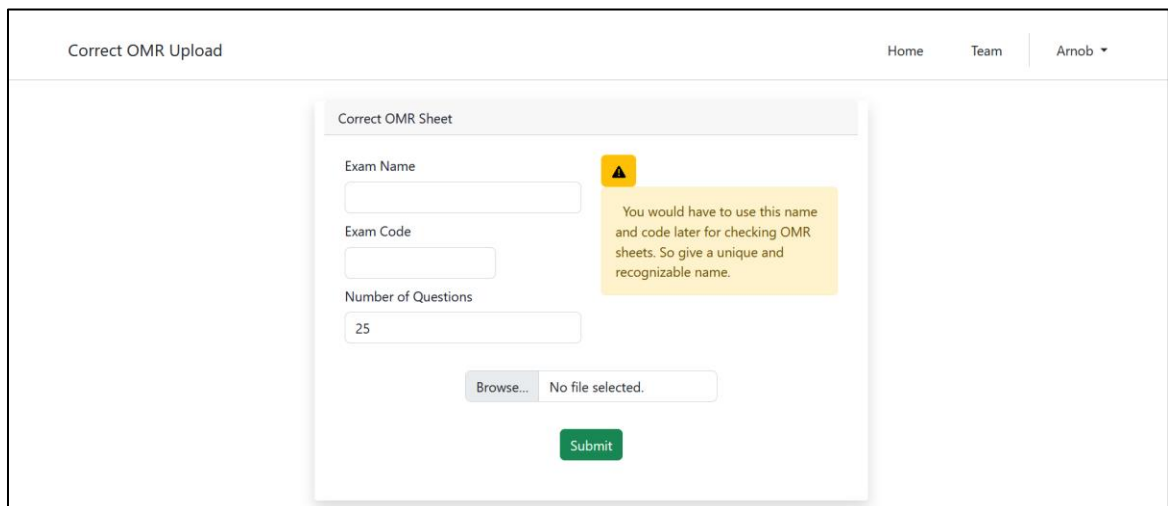


Figure 13: Correct OMR upload page

- **Customize correct OMR**

Customize UI OMR

Home Team Arnob ▾

1	A	B	C	D
2	A	B	C	D
3	A	B	C	D
4	A	B	C	D
5	A	B	C	D

Submit

Figure 14: Customize OMR page


- **Check OMR page**

Check OMR

Home Team Arnob ▾

Each OMR takes 1 second to run. Be patient.

Check OMR

Exam Name 

Exam Code

☐ Not special OMR

You can upload multiple files at a time

Browse... No files selected.

Submit

Figure 15: OMR checking page

- **Previous Exams**

Exams

Home Team Arnob ▾

All Subject

Exam Code	Exam Name	Show
137	Physics	show
136	Chemistry	show
345	Biology	show
102	Math-2	show

Figure 16: Previous exams page

- **Results of corresponding exam**

Result			Home	Team	Arnob ▾
Physics					
Student Roll	Obtained Marks	Details			
1807088	25	<button>show</button>			
1807082	20	<button>show</button>			
1807085	18	<button>show</button>			
1807083	14	<button>show</button>			

Figure 17: Exam result page

- **Result of individual student**

Detailed Result			Home	Team	Arnob ▾
Roll 1807082					
1	A	B	C	D	
2	A	B	C	D	
3	A	B	C	D	
4	A	B	C	D	
5	A	B	C	D	
6	A	B	C	D	
7	A	B	C	D	
8	A	B	C	D	

Figure 18: Individual result page

3.3 Performance Evaluation

In the case of our customized OMR sheet, the system maintains 100 percent accuracy, but when we check for any random OMR, it sometimes gives the correct solution but sometimes fails. Again, if we highlight the border, then it also works perfectly.

Chapter 4: Conclusion

4.1 Conclusion and future work

In our project, we have evaluated OMR sheets using image processing techniques. We have created our own customized OMR sheet and got the expected outcome from it. We also tested the OMR sheets of some institutions, and the results were not satisfactory in some cases. So, in the future, we want to improve our system so that it can work accurately for any kind of OMR sheet. As a result, we hope that the OMR sheet evaluator will be simpler to use and more accessible to all.

References

- Nikita Kakade, 2Dr.R.C.Jaiswal,OMR Sheet Evaluator using image Processing ,<https://www.jetir.org/papers/JETIR1712119>
- <https://docs.opencv.org/4.x/>
- <https://getbootstrap.com/>
- <https://www.w3schools.com/php/>