

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/369769816>

AUTOMATIC MULTIPLE CHOICE EXAMINATION QUESTIONS MARKING AND GRADE GENERATOR SOFTWARE

Article · April 2023

DOI: 10.12962/j20882033.v33i3.14522

CITATIONS

0

READS

34

7 authors, including:



[Benjamin Kommey](#)

Kwame Nkrumah University Of Science and Technology

64 PUBLICATIONS 93 CITATIONS

[SEE PROFILE](#)



[Eliel Keelson](#)

Kwame Nkrumah University Of Science and Technology

19 PUBLICATIONS 62 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



Attendance Systems [View project](#)



Intelligent Underground Mine-Rescue System [View project](#)

ORIGINAL RESEARCH

AUTOMATIC MULTIPLE CHOICE EXAMINATION QUESTIONS MARKING AND GRADE GENERATOR SOFTWARE

Benjamin Kommey* | Eliel Keelson | Frimpong Samuel | Seth Twum-Asare | Konadu Kwaku Akuffo

Department of Computer Engineering,
Kwame Nkrumah University of Science and
Technology, Kumasi, Ghana

Correspondence

*Benjamin Kommey, Department of
Computer Engineering, Kwame Nkrumah
University of Science and Technology,
Kumasi, Ghana. Email:
bkommey.coe@knust.edu.gh

Present Address

Departement of Computer Engineering,
Kwame Nkrumah University of Science and
Technology, Kumasi, 00233, Ghana

Abstract

This paper discusses a feasible software solution that enables automatic marking and grading of scripts. Technology keeps expanding, and more advanced innovations are being implemented with time. The marking and allocation of grades for examination scripts through human efforts are gradually becoming a thing of the past. Hence, machines and software applications are introduced to make the entire marking and grading of examination scripts more efficient, fast, and less tedious. Computer vision is an artificial intelligence (AI) knowledge domain that ensures devices obtain useful information from digital images, videos, and other visual inputs. Image processing and recognition, a unique part of computer vision alongside the python programming language and the OpenCV library was employed for this project. These are the most used in developing most recent applications that utilize, to some extent, artificial intelligence to attain specific desired results. The result of the project seeks to develop a maintainable android software application that uses image processing technology to scan patterns or images and grades results of multiple-choice question scripts based on a set marking scheme. This ensures that desired results are obtained while increasing efficiency and productivity.

KEYWORDS:

OpenCV, Computer Vision, Automatic Marking, Digital Image, Examination, Grading

1 | INTRODUCTION

Final year projects are incorporated into the academic curriculum of most universities to enable students to apply or demonstrate their skills and knowledge in solving real-life problems, become more resourceful, and serve as a guide in their career paths. In this modern age of technological advancements, some individuals or institutions still subscribe to marking objective/multiple choice examination sheets and grading them manually. This can be very demanding and stressful when many students participate

in the exams. There was great interest in undertaking this project since it is computer vision related. There is also a growing need for more AI and machine learning personnel in the computer and software engineering fraternity because of recent technological innovations more inclined towards these fields. Image recognition and processing, a sub-part of computer vision, is the base of this project. It inculcates OpenCV, a tool for image processing and performing computer vision tasks. It is an open-source library that can perform tasks like face detection, objection tracking, and landmark detection. It has many useful functions and algorithms for most computer vision tasks. Image recognition is the ability of a device-powered camera to identify and detect images, and image processing is the field of enhancing the images by tuning their parameters and features. The project would enable the use of our knowledge, research, and skills in programming with python and the use of OpenCV to assist in implementing an automatic marking and grading android software application that would be hugely beneficial and solves an academic-related issue.

Over the years, objective or multiple-choice examination sheets have been marked and graded solely on human efforts. This turns out to be tedious and demanding, especially since the student number is huge. Engineers then developed Optical Mark Recognition (OMR) machines that could take on the job and do it more effectively, reducing the burden on teachers, lecturers, and instructors. However, with the high production of software over the last decade, especially those implementing more complex algorithms and infrastructure, software applications that perform the same task as the machines are deemed possible to build and execute.

The advantage of utilizing this software is that they are very portable, easy to use, and subject to continuous change and improvement. This project seeks to invent an application for exam test grading in a fully automatic way, where we put into practice the resources that computer vision makes available. Examinations done in the multiple-choice questions set up are written by students using Optical Mark Recognition (OMR) sheets. On the OMR sheets, there are slots for questions to be answered with blank squares or bubbles corresponding to possible answers for which the student taking the exam can shade a blank square corresponding to their preferred answer with usually HB pencils. To mark the answers provided by students, educational institutions would have to use an OMR scanner which is a machine that detects the presence or absence of a mark and uses it to mark and grade student answers with at least a 90% accuracy concerning a provided marking scheme by the examiner. The OMR machine is substantially costly, and for that matter, most educational institutions do not have the financial muscle to spend in excess of \$10,000 to purchase and maintain an OMR scanner and hence tend to outsource the marking process to firms that provide the services of OMR sheets marking at the cost of 10 cents per page (which is still costly) or utilize the time of teaching assistants (and in the case of most Junior High Schools, "trusted students" also give a helping hand to their teachers) to help in the manual human-error-prone marking of the answer sheets. This project would help educational institutions, especially financially constrained ones like Primary Schools, Junior, and Senior High Schools, to gain access to powerful OMR sheets marking technology with at least a 90% accuracy (rectifying the human error issues with manual marking that could have momentous effects on students) at a zero cost (not having to buy and maintain an OMR scanner or outsource marking at a fee).

This project sets out to implement an AI technology to mark and grade OMR sheets used in multiple choice question examinations with at least a 90% accuracy. To do this effectively, gathering data and processing it for training and testing is necessary. This includes acquiring sample OMR sheets, preprocessed to convert them from grey images into a binary image that is a multidimensional array formatted into a uniform size to represent the image. There would be a way to detect multiple shading for questions and questions left blank. Other parameters, such as the index number and course code, would also be retrieved and used to present the data in a RESTful API built on top of the AI module to be developed. Many lecturers and teachers find marking answer sheets and subsequently having to correct errors on answer sheets very boring and primitive. Therefore the need for this free and open-source technology would automate the marking procedure and remove any human errors from the process.

2 | PREVIOUS RESEARCHES

Several automated formative assessment options are widely accessible and commonly used in learning management systems and grading students today. Students can take tests generated from a vast pool of questions electronically, which are then automatically assessed and provided with feedback^[1] In this section, various design implementations and features of the proposed system have been described in detail. Reviews of other mark and grade systems in the literature have been summarized. These works' various contributions and/or weaknesses have been focused on to help contribute to measures taken to tackle the proposed system's issues or drawbacks of the technologies discussed.

Talib et al.^[2] investigates and proposes a shape-based vision algorithm, a hierarchical template-matching approach implemented in this system to verify the imaging and inspect the correct answer of the Optical OMR sheet form, despite the use of a webcam in his implementation being undesirable. An OMR response sheet scheme would be used as a template for object recognition during the matching process, with all right answers written on the paper. To extract the object's contour, a region of interest (ROI) is picked and filtered into a grey level using image processing techniques.

Gyamfi and Missah^[3] worked on an unsupervised model in MATLAB. Regarding speed and accuracy, their research aimed to classify the proposed pixel-based unsupervised classification OMR algorithm as either favorable or undesirable. According to their findings, the suggested OMR method was faster and more accurate than object-based unsupervised or supervised classification OMR algorithms. The cost, speed, and accuracy of the OMR algorithm were investigated using three algorithm development modules: template creation, picture preprocessing, and content categorization. Their employment of an unsupervised classification method resulted in a faster and more accurate result. Their work, however, might be improved to be implemented on a cell phone rather than a computer. Developers can improve the algorithm so that it can be used in mobile applications, such as capturing OMR sheets with a smartphone camera and processing them for grading.

Hussmann and Deng^[4] Created a high-speed hardware OMR system prototype for marking multiple-choice questions using a Field Programmable Gate Array (FPGA). It reached a high processing speed thanks to the use of the FPGAs. Effective mark detection and verification algorithms were developed and implemented using FPGAs to achieve real-time performance at a minimal computational cost. The OMR could process a 3456-pixel high-resolution CCD linear sensor at 5000 frames per second at the sensor's effective maximum clock rate of 20 MHz (45 MHz). Hardware solutions are quite expensive for the average institution in our communities, even though the installation was a low-cost technique.

Chai^[5] Document scanners could be used alongside desktop computers to perform the marking instead of relying on expensive, specialized, restrictive answer sheets and optical mark recognition scanners. While compiling the assessment results for the assessor, it can also annotate the scanned answer sheet images with feedback and send them back to students via email. Experiments reveal that the proposed method can mark a sheet in as little as 1.4 seconds with 100% accuracy. This looks promising but needs to be more affordable because document scanners are costly, and the extra spreadsheet software needed to be written for the program is costly^[6]

Patole et al.^[6] implemented a framework based on digital scanners and cameras, offering greater flexibility and allowing users to expand functionality. The framework uses a combination of Octave scripts and spreadsheet software to perform optical mark reading and automatic scoring. Nonetheless, this framework is limited by its reliance on custom-printed answer sheets.

Catalan^[7] tries to unravel OMR marking with Hough transform and skew-correction into the proper orientation, followed by the normalization to a given size. Next, the tick mark corresponding to the response for each question can be recognized by the issuance of the mask, which envelops the response area. The empirical outcomes showed that Nguyen et al.^[8] system had accomplished substantial advancement in performance in terms of exactness, dependability, and elapsed time likened to those of the conventional optical mark recognition (OMR) systems. The system also indicated high exactness of 99.7% while using non-trans optic response sheet paper with a lower expense. Auto evaluation of the OMR answer sheets project was done by Sinha and Gupta^[9]. Their objective was to use the OpenCV library for Android to create a simple application to mark answer sheets. Their target audience for the project was teachers, schools, and individual professors who could not afford or access the premium services of a dedicated OMR machine. Their app was simple and had three buttons. The capture reference sheet button was for capturing and uploading the reference answer sheet. The Capture student sheet button did the same thing but was for the answer sheets of the students to be assessed. After the camera has done its job, an image processing algorithm is run in the background to correct the tilt and rotation, if any, and to extract answers and store them in the list data structure. The last button, the view results button, compares the two data and counts the number of matching answers. The count is returned as the grade. Its strengths are its relatively simple interface with a fast processing algorithm that takes 0.8 seconds to process captured images. Weaknesses include it does not recognize students' identification and only one sheet can be graded at a time^[8]

Ascencio et al.^[10] created computer software for grading multiple choice as a cheaper alternative to the industry standard OMR. Python and OpenCV were used for algorithmic design, programming, and image processing. The template used was a predesigned test answer sheet, and the program could grade several papers simultaneously. The program could detect the ID number of the students and the test answer sheet. This was great for effective data categorization and matching grades to

respective students. The interface allows the user to use two buttons to upload the reference and the student's answer sheet (multiple times). A third button would start the image processing and grading algorithm. The algorithm inverted the colors of the paper. It used rectangular detection to detect the rectangles containing four circles based on the rectangle's width, height, and aspect ratio. The inversion of the image means that shaded answers are white and can be detected and counted by the algorithm. This software allows multiple sheets to be graded at once, but there is no android implementation of the application and no flexibility as answer sheets are fixed.

Kulkarni et al.^[11] worked on this project in 2013, intending to replace a dedicated OMR with a regular scanner and computer setup. The software also had GSM functionality due to a GSM modem used in collaboration. This meant that the software, after grading, could send the results to the students directly. A scanner scans the test sheet in a folder in .jpeg format. The software retrieves the images, and with OpenCV image processing, the answers are extracted from the images. They are graded and stored in a database. Here, the administrator can distribute the answers to the students via SMS. Strengths are the format of the answers is not fixed, and grades can be readily distributed. However, the setup is not compact and involves separate components. Scanning is also done manually, and paper alignment is not done automatically. Multiple-choice questions have become a crucial part of the academic system. Necessary exam tests also use multiple-choice questions to evaluate students' academic performance. The Indian Institute of Technology (IIT) examinations are just some of the many important tests operated in India to get the students on a joint stage. Every year millions of students take these tests, and they must answer various questions asked by emphasizing the circles in OMR sheets. This exam operating institution uses a technical machine for grading MCQ paper-based exams. It is very costly and needs a specially oriented operator to operate the machine accurately and efficiently. These colleges use expensive OMR software and the machines associated with assessing the OMR sheets.

So, our hypothesis is to formulate a procedure to use a personal computer plus a scanner and a program-based application that would grade a specially designed MCQ exam test paper with questions with four choices for each question, and the student can choose only one answer per question. The program would test to glimpse the correct answers by comparing each paper with a correct answer which would be pre-stored in the database. The program is written so that it can accept rotating the papers in the scanning operation using the image registration process. The program would use a simple software application such as OpenCV, a library of programming functions to implement the image-processing algorithm, SQL for database, and Visual Studio, in which program code would be developed. IDE (Integrated development environment), too, is a software application with facilities for computer programmers for software evolution. The system mandates very less storage space, so it has no storage problem. It has a fast-processing speed. Not portable, sheets must be in black and white and require a scanner (external)^[10]

Automated electronic processing of test results has yet to be recent. It is utilized in almost all national standardized testing performed by the national administration in public and private schools, such as the National Achievement Test (NAT), National Career Assessment Exam (NCAE), Basic Education Exit Assessment (BEEA), and others^[6]. There are efforts to automate the processing of test results locally. For instance, the City Schools Division of Santa Rosa City has obtained an OMR reader^[6]. According to the same author, it aligned with their commitment to using information communications technology. Another spirited endeavor to execute the same automation came from the Division of Batangas City. In 2015, the Division of Batangas City purchased ten (10) OMR machines worth nearly twelve (12) million pesos through their connections with the local government.

According to the same author, the action was in line with the advocacy of the Department of Education towards a shared insight into the objectives of quality education, learning, and objective evaluation. OMR is a method that compiles data by identifying marks on paper. A hardware device achieves OMR by noticing a reflection or a limited portion of light. OMR readers or scanners are normally used to scan forms/sheets where an examinee would use a pencil or ball pen to shade in a circle or other geometric shape on a form to answer a question. The publications on OMR machines, on the other hand, have produced optimistic results and favorable recommendations. Nonetheless, the apparent drawback is that OMR machines are quite expensive. It was also confirmed, as noted in the studies of Catalan^[7], Patel and Prajapati^[12] that OMR machines are expensive to achieve and conserve. To make something out of it, an option is necessary. An option that would qualify for such advantages but has no disadvantages. Smartphones can provide a much better and more reasonable alternative in integrating automation into the processing of the test outcomes of students. Smartphones' digital cameras and image processing software offer this cheaper alternative. As an option, it should satisfy the necessity for a cheaper option and provide functionality within the leaps of outstanding quality^[11]

A research paper drafted by Catalan^[7] from the Department of Physics, Tsinghua University, Beijing, China, in 2019 for traditional automatic grading methods, many of them are based on optical mark reader (OMR). Therefore, they can locate the answer areas accurately. Nonetheless, these procedures rely on professional photographic equipment, which could be more suitable for the users^[12]. In recent years, some works have introduced image processing techniques to the OMR-based methods^[8, 13, 14]. With the evolution of smartphones and their in-built camera technology, instead of relying on specific equipment for scanning, some automatic grading methods established on smartphone pictures have been proposed^[15, 16]. In this case, students and their teachers can get the grading results instantly without extra equipment. We propose an automatic homework grading system that utilizes answer area detection, character recognition, and grading through the execution of algorithms such as the Harris corner detection. It must be simplified since it integrates many algorithms and models^[17].

Darvishzadeh et al.^[18] and Muangprathub et al.^[19] proposed an automatic system to grade multiple choice questions to detect the correct answers by comparing each paper with a pre-scanned test paper that contains the correct answers, as many forms of test papers are used in real exams conducted in the computer center of the Baghdad University. The results matched results from the manual grading of the same papers. It involves image processing and template matching^[20–23], which is a character recognition technique. It is the process of finding the location of a sub-image, called a template, inside a larger image. Once a few corresponding templates are found, their centers are used as corresponding points^[24]. The best similarity score window is selected as the match location. The answer sheets are scanned by the scanner to generate images. These images are converted to binary images, template-matched, and exported. The motivation for automatic image-based grading of multiple-choice answer sheets includes significant time and cost reductions. The proposed system applied the correlation coefficient to compare answer sheets and solution sheet images. The proposed method supports any pencils and pens for marking the answers and supports thin or low-cost gridded paper as answer sheets, which are easy to use in a general test^[25].

This detection method was developed by Loke et al.^[26] in conjunction with a new forms processing application that implements the software OMR^[27]. It was found during preliminary field testing that the standard methods of mark detection needed to be more accurate when used on forms filled by untrained respondents. The equipment used for printing and scanning was heavily used, resulting in many scanned forms having artifacts of some sort. Finally, there was a need for internal labels for bubble input fields as the bulk of our survey forms used these fields due to the higher SOMR accuracy compared to handwriting fields. Questions with internal labels were typically 30–50% more compact than external ones, allowing us to print forms with fewer pages. This detection method consists of three sequential processes: removal of internal labels, basic mark detection, and correction of print and scan artifacts. Removing internal labels is done by masking out the labels with a template so that any residual pixels can be attributed solely to user marks. Mark detection involves simple pixel counting after preprocessing to remove noise. Finally, common print and scan artifacts are corrected using a statistical technique isolating user marks from the artifacts. It is entirely software-based and highly accurate due to detection techniques such as adaptive X-mark matching^[26]. Compared to hardware-based OMRs, scan speeds are slower^[28].

Čupić et al.^[29] proposed a solution that enables automated attribution of formative exams to students. We define the student identifier matrix as a graphical representation of a student ID number. The student annotates the matrix during the exam. We introduce a robust image processing procedure that enables automated recovery of the student ID from the matrix. We show that the proposed procedure achieves a recognition rate of 100% on 296 samples in the presence of skewing, rotation, and offsets introduced by the printing and scanning processes. The student identifier matrix is a table of 10 rows and the number of columns equal to the length of the student ID number. Each column represents possible values of one digit of the student ID number. The number of rows is ten, with ten possible digit values (0-9). The idea is that the student annotates a cell in each column corresponding to the appropriate digit in her student ID. To make using the student ID matrix easier, each cell is preprinted with its corresponding value. We use several image-processing techniques to detect the student identifier matrix. The student identifier matrix is printed on the exam form, and the approximate location of the matrix is known in advance. However, the precise location must be verified because printing and scanning can introduce some rotations and offsets of the complete exam form. Considering that the matrices would be used on exams in courses with many students, we require the detection algorithm to operate very fast. In addition to scanning the student ID number, the student's solutions for the exam also need to be scanned and processed. The process of detecting the student ID matrix begins with manual annotation of the approximate area of the matrix on a single exam form sample. It involves complex matrix formula but has a recognition ratio of 100% and is, therefore, very reliable^[30].

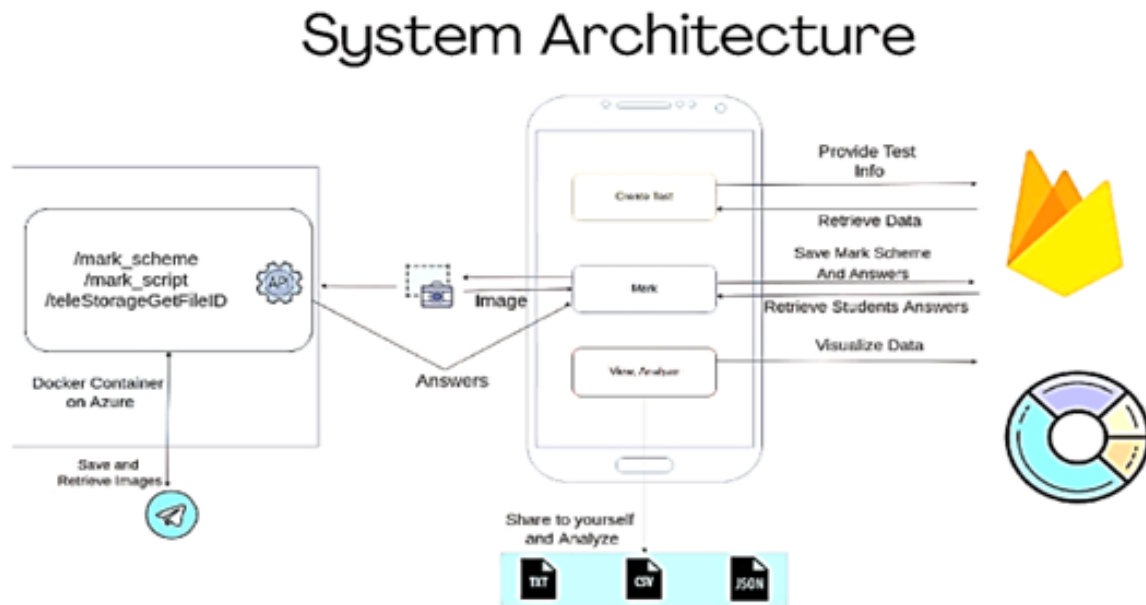


FIGURE 1 The system architecture of OMR machine.

The main aim of this work by Nirali V Patel and Ghanshyam I. Prajapati in 2015 is to completely remove the ordinary scanners by using a web camera as an input device of the OMR sheet. An OMR sheet is placed before a webcam, and the program takes its image. It used camera-based OMR for devices with multi-core processors and was developed using OpenCV libraries. First, the image would be the threshold then the bubble location is found in the answer sheet. Each pixel in the image is separated as an object or a background. Adaptive binarization^[12] and bubble detection are implemented. The gray level difference between marked and unmarked bubble is the main feature in the classification process. The difference in the gray level of the current bubble and the background is used to trim back the effect of noise and illumination variation. Thresholding is the simplest form of image segmentation used to create binary images. Image gridding involves making a grid. The grid would be drawn over the image so that each square or rectangle contains an optical mark or a black dot. After this, we distinguish each black dot according to the rectangle in which it is contained. The software is multi-functional because it has been developed with extensive OpenCV functions and algorithms and may be slower than hands-on OMR machines.

3 | MATERIAL AND METHOD

3.1 | System Architecture

The system architecture is depicted in Fig. 1. A couple of OMR sheets are answered by candidates taking part in tests or examinations. The answering is done by shading their supposed right answers. The sheets are gathered, and the camera of an android smartphone then takes images. The android device in question has the application installed on it. The examiner is the only individual with authorized access to the marking/grading of the examination sheets at that time. The examiner installs, registers, and logs into the application. An image of the reference OMR sheet/marketing scheme is stored as the master template. After that, images of the answered sheets by examination candidates are taken, preprocessed, and compared to the master template to generate scores or grades.

The results are then subject to data analysis and database storage. If an OMR sheet is blank, it would still undergo preprocessing, but since there is no data to retrieve, processing would stop, and the sheet would reject or deemed as an error. Activities on the application would be synced or backed up with the examiner's account so they can still access information when they log into the application on another android device.

3.2 | System Block Diagram

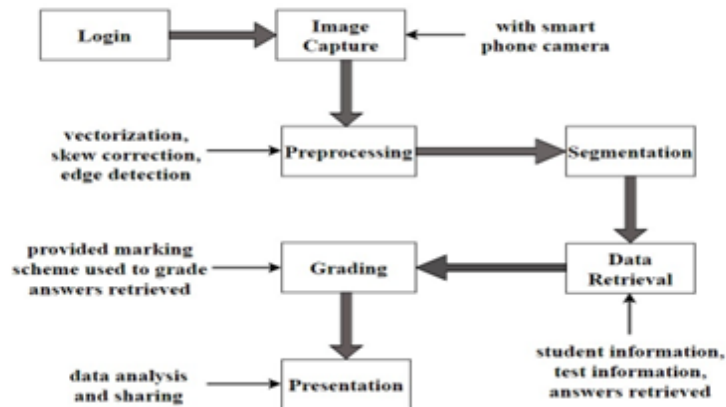


FIGURE 2 The system block diagram.

Fig. 2 shows the system block diagram. For the preparation phase, the examiner logs into the application and takes an image of the reference OMR sheet. This image undergoes preprocessing, feature extraction through modular recognition, and data retrieval. The results obtained after data retrieval are saved as the master template. Images of the examination candidates are captured and taken through preprocessing. Preprocessing is making the image ready for data retrieval. It involves vectorization, normalization, skew correction, and noise reduction. Vectorization is converting the image into its matrix representation.

Normalization refers to changing the range of pixel intensity values of the image so data retrieval can be efficient. Skew correction ensures there would be a high level of accuracy in preprocessing irrespective of the image alignment or tilt. Noise reduction ensures the number of redundant marks involved in preprocessing is significantly reduced. Feature extraction through modular recognition is next. It ensures each section on the OMR sheet, such as student info, test info, and answers, are treated as modules and worked on independently and separately. Data is then retrieved and compared to the master template to generate scores. The results can then undergo data analysis to determine to mean, highest or lowest scores. They could also be stored in a database for subsequent review or use. Email of results can also be sent to the candidates.

3.3 | System Software Workflow

As illustrated in Fig. 3, the system workflow gives the flow of procedures in the application. It shows the start, conditions, and feedback. After the software opens and there is access to the homepage, there is the option to start new grading or test. A new table would be created in the database for further processes. The examiner could now take images of the reference answer sheet, extract or retrieve answers using algorithms and store them in temporal data structures such as list representations. The same processes are repeated for the candidates' examination sheets. The total number of matching answers becomes the grade of that candidate's answer by comparing the answers of the reference and candidates' examination sheets in data structures. The candidates' IDs and grades are then stored in a table. The table is displayed, so the examiner can view it and then have options to share, review, or perform data analysis on it.

3.4 | System Software Design

The system software was designed using various programming languages, frameworks, and software packages. Python programming language, OpenCV, and Flask for Image Processing were used for the software design. Python is the programming language used for writing the algorithms and functions deployed in this application. OpenCV is a library of computer vision functions, originally developed in C++, and OpenCV- Python is a python wrapper for OpenCV's python bindings. This project used OpenCV- Python to design and architect the computer vision models on Google Collaboratory. Google Collaboratory's GPUs and free runtime of 12GB of RAM enabled us to experiment with various techniques and use-case scenarios.

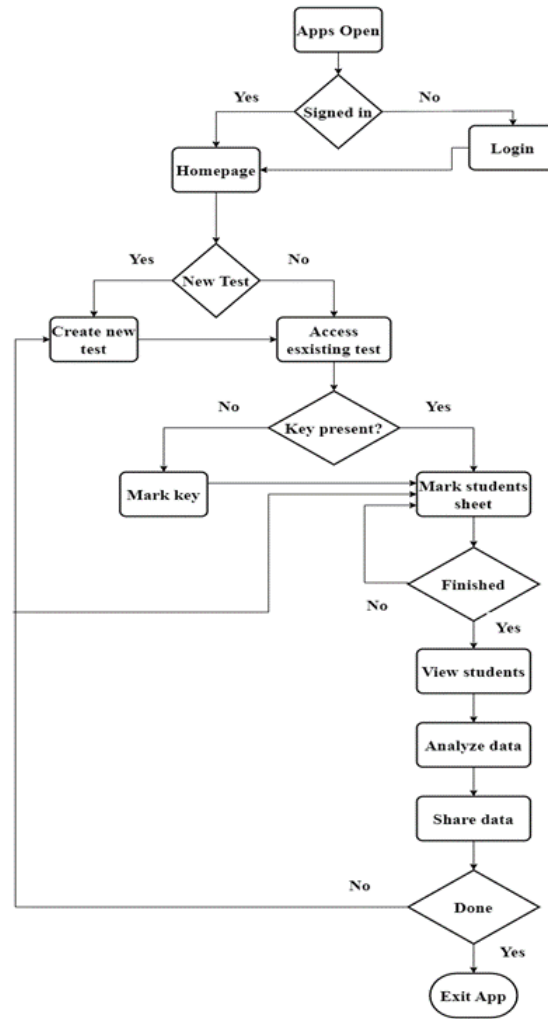


FIGURE 3 The software workflow.

Flask is a lightweight web framework for python meant to make development easy and seamless. Flask was used to wrap the computer vision models into a REST API that the software and other applications can consume. With the open API, other developers from other institutions can clone and tailor it to their requirements. Firebase-Firestore was used for user account handling and data management. Firebase is an app development platform that allows you to create and grow popular applications. It has a wide range of services, from hosting, real-time database, analytics, in-app messaging, push notifications, and many more. Backed by Google and trusted by millions of businesses worldwide, we decided to use the platform's service called Firestore, a NoSQL database solution for developing cross-platform applications. Firebase-Firestore provides authentication through various means to prevent unauthorized access to data. The Firebase Firestore is shown in Fig. 4 .

3.4.1 | Telegram API for Cloud Storage

Considering the cost of cloud storage services from AWS to GCP in our early stages of production, we decided to be a bit hacky. We tweaked telegram's open-source cloud-based API services to serve as cloud storage for our application. With this cloud solution combined with firebase-firestore (which would be discussed next), captured images of answer scripts could be uniquely identified, stored, and retrieved at the cost of \$0.00 forever.

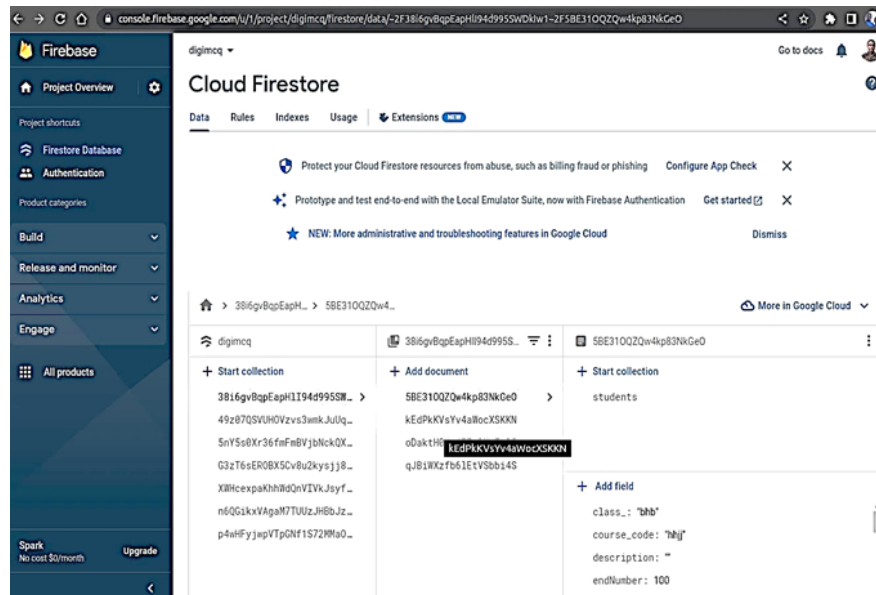


FIGURE 4 The firebase cloud firestore.

3.4.2 | Edge Detection

Using OpenCV-Python to implement an edge-detecting algorithm, there was a bit of a challenge: which was the offset that was shown when the detected edges were drawn on original images and given the fact that our projects deal with a refined scale (to the nearest 1/4th of a millimeter), it was necessary to find an edge-detecting algorithm that solves our challenge. Edge Detection is a dart package that solves our challenge by enabling the user to crop a captured image even after the AI has detected the edges. It's easy to integrate into our flutter application since it's written in the dart, the language used to write flutter. Fig. 5 depicts the Flutter edge detection plugin.

3.4.3 | Flutter for Mobile Applications

After wrapping our module into a REST API that could be consumed from an HTTP request, we needed to develop a user-friendly mobile application for our application. Flutter is an open-source cross-platform UI Software Develop Kit developed by Google. It enabled us to develop a simplistic UI to connect to our API and databases seamlessly.

4 | RESULTS AND DISCUSSION

The individual software components and well as the complete system was tested. A test was performed to check for accuracy and system robustness. To begin with, they were placed on a table to scan and detect edges. Fig. 6 shows the initial edge detection stage. Some screenshots, descriptions, and functions of various components and test results are as follows.

Login, Sign Up, and Home Page

The page where logging into the application or registration takes place. It displays the various examinations or tests and a tab to create a new test. The screenshots are displayed in Fig. 7 and Fig. 8 .

Creating New Test

It provides placeholders to enable the examiner uniquely to describe each test or examination. The create a new test screen is shown in Fig. 9 .

Marking Key/Master Template and Candidate Scripts

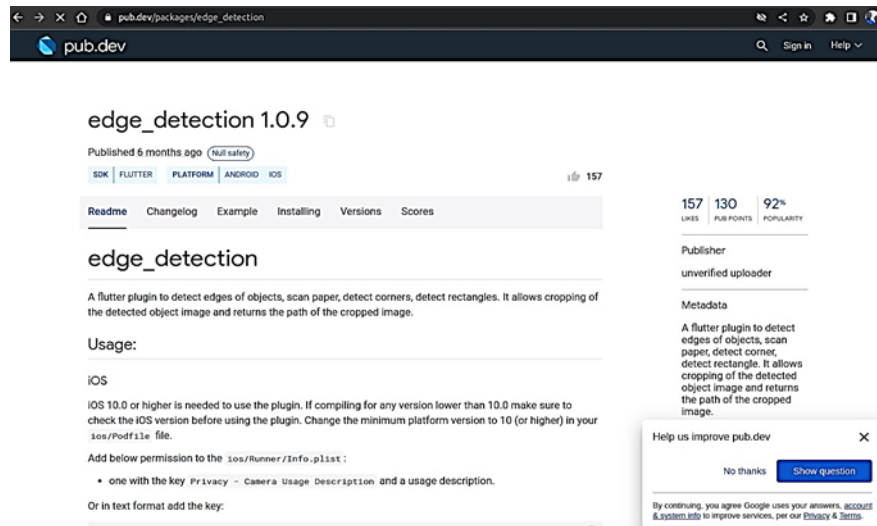


FIGURE 5 The flutter edge detection plugin first view.



FIGURE 6 The edge detection initial test.

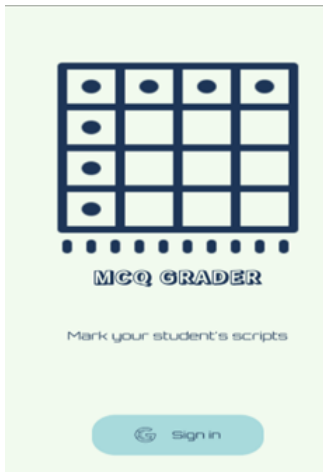


FIGURE 7 The log-in screen.

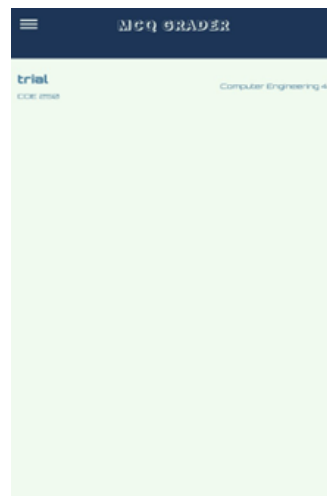


FIGURE 8 The home screen.

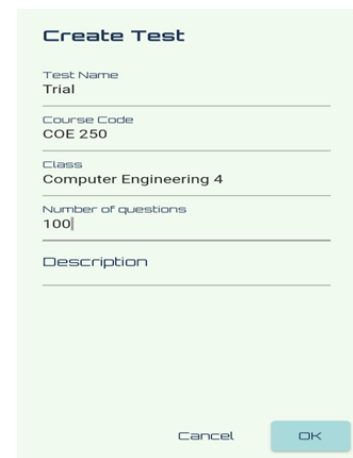


FIGURE 9 Create a test screen placeholder.

Pressing the key at the bottom right corner opens the camera to take an image of the marking scheme or master template. Likewise, pressing the marked script at the bottom left corner opens the camera to take an image of the candidates' scripts. The screenshots are shown in Fig. 10 and Fig. 11. After marking and grading, results are compelled and displayed, as illustrated in Fig. 12.

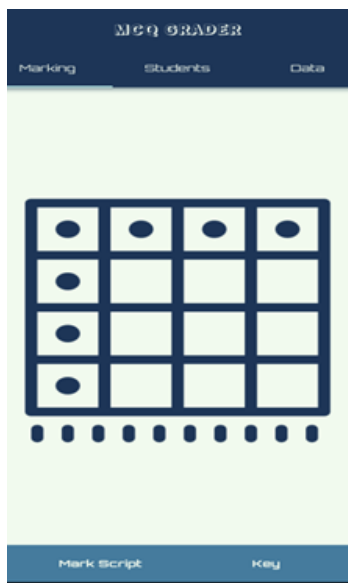


FIGURE 10 Before scan capture screen.

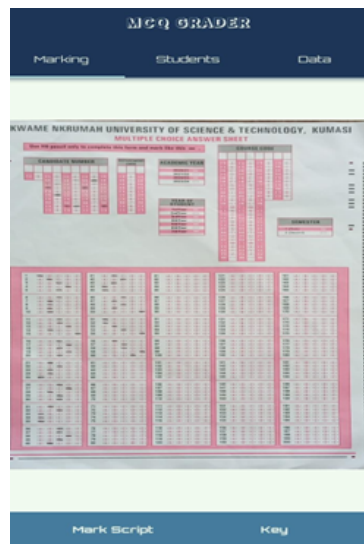


FIGURE 11 After scan capture screen.

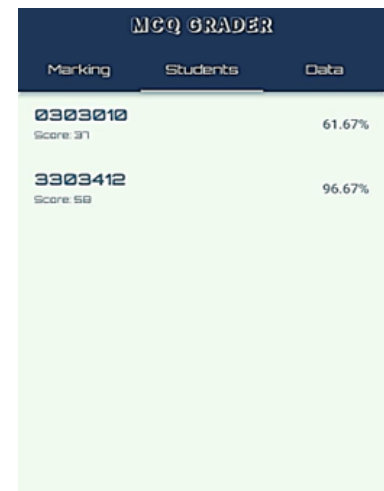


FIGURE 12 Results displayed after grading.



FIGURE 13 The export screen.



FIGURE 14 The sharing screen.

Data analysis can be performed after the results are displayed. The data section or page separates the grades into A, B, C, D, and F categories in a pie chart-like form, and the results can finally be exported through text, CSV, or JSON, depending on preference. The screenshots for data analysis, presentation, and exports are shown in Fig. 13 and Fig. 14.

Google Collaboratory

Google Collaboratory, or Google Collab for short, is one of Google Research's products. It is useful for machine learning, data analysis, and education because it allows users to combine executable code with rich text in one document, which can then be run from the browser. They are stored on a user's Google Drive account and can be run from any device that the user authenticates. Fig. 15 shows the integration of the Google Collaboratory into the system software application.

After the API was developed, there were attempts to deploy it onto a cloud service and host it. Still, there needed to be more bottlenecks regarding operating systems, versions of dependencies, and other variables. Anytime there was a change in the code,

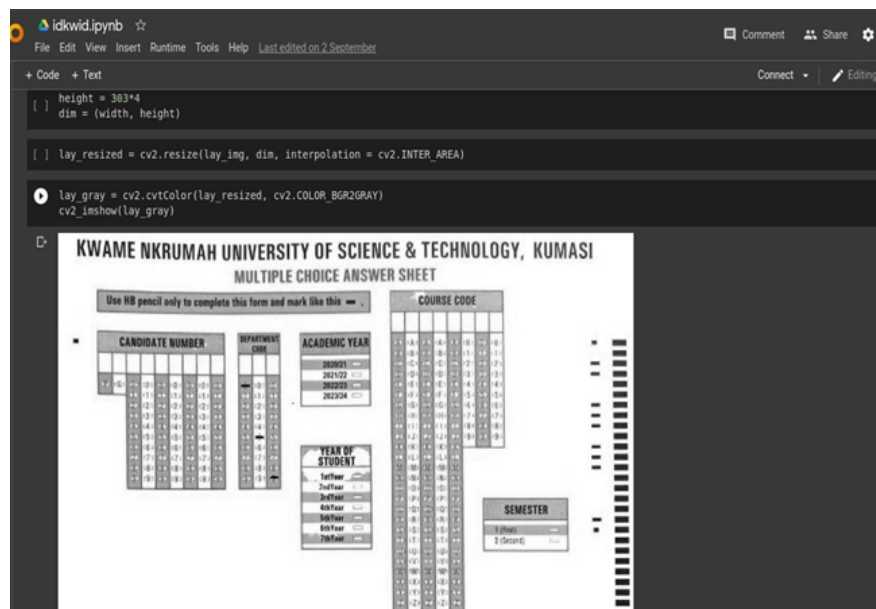


FIGURE 15 The Google Collaboratory.

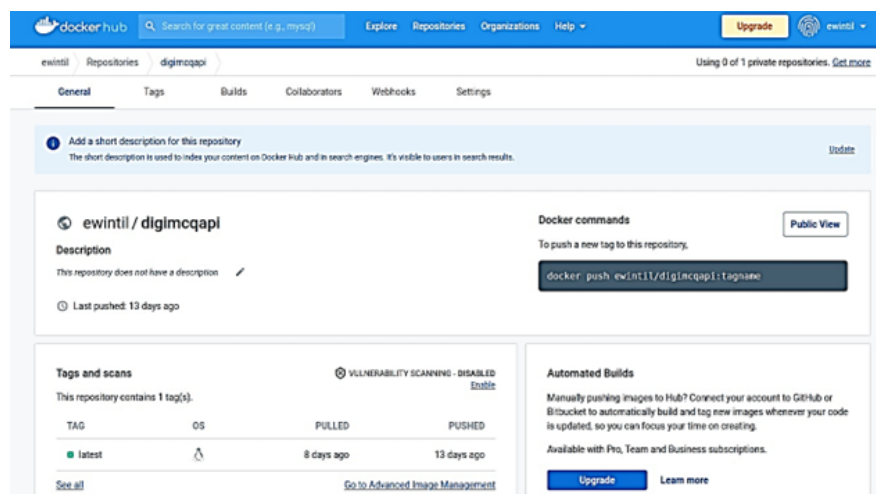


FIGURE 16 The docker first view.

it had to be redeployed and possibly made changes to the cloud platform to accommodate the changes. Docker is a service that uses OS virtualization to deliver software in packages called containers. With these containers, the software can be deployed without worrying about the cloud provider's operating system and dependency version conflicts. Deployment is just a push away. So, the system API was bundled into a Docker container and deployed on Microsoft Azure, a cloud computing service just like Amazon Web Service or Google Cloud Platform but operated by Microsoft - it ensures that our API server serves its users, in this case, the MCQ system mobile software application and stays online even if the developers go offline. Fig. 16 and Fig. 17 depict the docker and the Microsoft Azure interfaces. While Fig. 18 shows a sample shaded scannable, and Fig. 19 shows a screenshot of indexes and grading scores.

It was noted that the system software application could retrieve the data in the images, make meaning out of it and generate results for the respective candidates' scripts. Multiple accounts and sheets were used to test the software at varying periods in time. Image in Fig. 19 shows two candidates/students, their index numbers, and their respective grades.

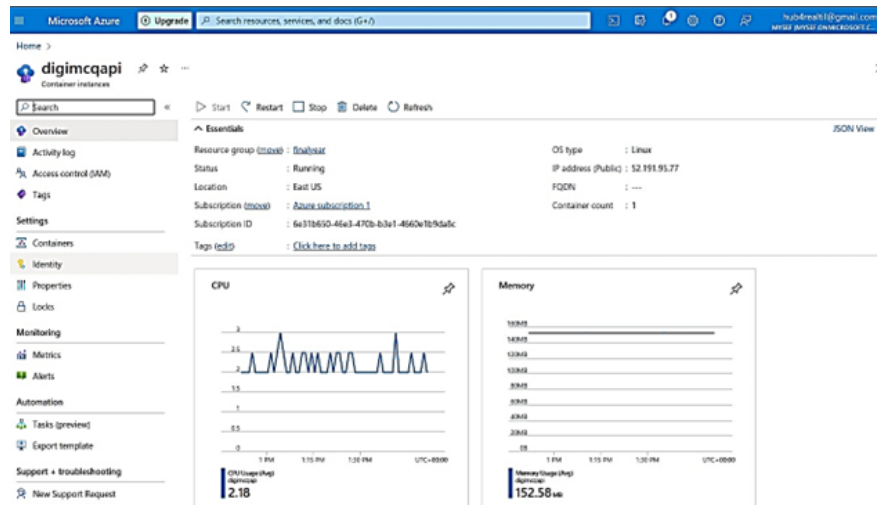
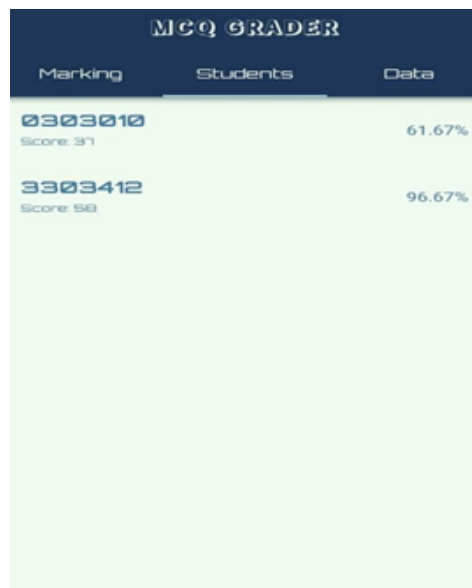


FIGURE 17 The Microsoft Azzure.

FIGURE 18 The sample MCQ Script used for testing..

5 | CONCLUSION

The project built a mobile software solution to automatically mark optical mark recognition sheets for school objectives or multiple-choice questions. The backbone of the project, the processing layer that processes the captured images and marks



MCQ GRADER		
Marking	Students	Data
0303010	Score: 37	61.67%
3303412	Score: 58	96.67%

FIGURE 19 The image shows index numbers and grades.

the sheets, has been open-sourced. Hence, developers can tailor the models to suit their needs without licensing. Beyond the objectives of this project, we've also demonstrated how to combine a NoSQL database like Firestore and telegram's open API to architect a free cloud storage solution that can be integrated into any software application. Considering the limitations of the application, we recommend that users of our software application and developers who would reproduce or enhance our development use the application in a room/environment with adequate lighting to preserve the sharp contrast between marked and unmarked regions. The algorithm works best when the difference between the marked and unmarked regions is appreciable; good lightning would amplify such difference and better results. Also, users should place the OMR sheet on a darker background when capturing the images. Doing so makes the edge detection process easier and reduces the time to capture and detect edges. As a result of inadequate diversity in the sample sheets used to train our algorithms, our algorithm only works for KNUST objective choice questions. Hence, developers are advised to tailor the parameters for training the models to suit the optical mark recognition papers that may be developed.

Moreover, though the app was developed and could work on both commonly used mobile platforms (Android and iOS), it was built and tested for only Android devices considering its popularity, ease of usage, and straightforward build and deployment process. Developers are encouraged to build and test the application on physical iOS devices before deployment. The quality of the camera of the user's phone limits the accuracy and precision of the application. Therefore, the application user needs to ensure that the camera of the phone using the application is of reasonable quality, where reasonable implies it would be able to capture the small sharp shapes of the OMR sheets. The application's architecture relies on regular API requests to the hosted server for marking and retrieval of data. This introduces an extra network-related factor that could become a bottleneck considering the slow network speeds of our cities in Third World countries such as Ghana. As such future developments would "shift" the cloud processing of the images into the application itself, and hence all the processes involving the application would be done in-app. Finally, it is recommended that future developments improve the design of the architecture to make the marking of the OMR sheets distributed: such that a user can authenticate other users to mark OMR sheets for him for a given period. Currently, the design is linear and doesn't support a distributed way of marking, and as such, the marking process is done one after the other. Still, a distributed system would enhance the user experience and increase the script marking speed.

CREDIT

Benjamin Kommey: Conceptualization, Methodology, Writing - original draft preparation, and Supervision. **Frimpong Samuel:** Formal analysis and investigation. **Seth Twum-Asare:** Writing - review and editing. **Konadu Kwaku Akuffo:** Resources, Drawing, and Editing. **Eliel Keelson:** Methodology, Writing - and Supervision.

References

1. Karavirta V, Korhonen A, Malmi L. On the use of resubmissions in automatic assessment systems. *Computer Science Education* 2006;16:141–157.
2. Talib A, Ahmad N, Tahar W. OMR Form Inspection by Web Camera Using Shape-Based Matching Approach. *International Journal of Research in Engineering and Science (IJRES)* ISSN 2015 4;3:29–35. <https://www.ijres.org/papers/Volume%203/v3-i4/D03042935.pdf>.
3. Gyamfi EO, Missah YM. Pixel-based unsupervised classification approach for information detection on optical markup recognition sheet. *Advances in Science, Technology and Engineering Systems* 2017;2:121–132.
4. Hussmann S, Deng PW. A high-speed optical mark reader hardware implementation at low cost using programmable logic. *Real-Time Imaging* 2005;11:19–30.
5. Chai D. Automated marking of printed multiple choice answer sheets. In: 2016 IEEE International Conference on Teaching, Assessment and Learning for Engineering IEEE; 2017. p. 145–149.
6. Patole S, Pawar A, Patel A, Panchal A, Ravindra J. Automatic System for Grading Multiple Choice Questions and Feedback Analysis. *International Journal of Technical Research and Applications* 2016 3;39:16–19. www.ijtra.com.
7. Catalan JA. A Framework for Automated Multiple-Choice Exam Scoring with Digital Image and Assorted Processing using Readily Available Software. In: DLSU Research Congress 2017 De La Salle University; 2017. p. 1–5. <https://www.dlsu.edu.ph/wp-content/uploads/pdf/conferences/research-congress-proceedings/2017/HCT/HCT-II-026.pdf>.
8. Nguyen TD, Manh QH, Minh PB, Thanh LN, Hoang TM. Efficient and reliable camera based multiple-choice test grading system. In: *International Conference on Advanced Technologies for Communications IEEE*; 2011. p. 268–271.
9. Sinha SK, Gupta PK. Auto Evaluation of OMR Answer Sheets Using Mobile Application. *Integrated Research Journal of Management Science and Innovation* 2015;6:12–17. www.irjmsi.com.
10. Ascencio HE, Peña CF, Vásquez KR, Cardona M, Gutiérrez S. Automatic multiple choice test grader using computer vision. In: 3rd IEEE Mexican Humanitarian Technology Conference, IEEE; 2021. p. 65–72.
11. Kulkarn A, Mane A, Kadam L, Sonawane M. Automatic Evaluation of Multiple Choice Questions with the Relaying the Result using GSM Network. *International Journal of Infinite Innovations in Technology* ISSN 2017;5:2278–9057. <https://ijit.logicinside.net/v5i4p05/>.
12. Patel NV, Prajapati GI. Various Techniques for Assessment of OMR Sheets through Ordinary 2D Scanner: A Survey. *International Journal of Engineering Research & Technology* 2015 9;4:803–807. <http://www.ijcaonline.org/archives/volume68/number13/1163>.
13. Smith AM. Optical mark reading - Making it easy for users. In: *ACM SIGUCCS User Services Conference ACM*; 1981. p. 257–263.
14. Chinnasarn K, Rangsanteri Y. Image-processing-oriented optical mark reader. In: *Applications of Digital Image Processing XXII SPIE*; 1999. p. 702–708.
15. Bayar G. The Use of Hough Transform to Develop an Intelligent Grading System for the Multiple Choice Exam Papers. *Karaelmas Fen Müh Derg* 2016;6:100–104.
16. Alomran M, Chai D. Automated Scoring System for Multiple Choice Test with Quick Feedback. *International Journal of Information and Education Technology* 2018;8:538–545.
17. Virtus L. Performance Mapping of Optical Mark Recognition (OMR) Machines in the Division of Batangas City: Basis for Program Improvement. *Ascendens Asia Journal of Multidisciplinary Research Abstracts* 2019 1;3.

18. Darvishzadeh A, Entezari N, Stahovich T. Finding the Answer: Techniques for Locating Students' Answers in Handwritten Problem Solutions. In: 16th International Conference on Frontiers in Handwriting Recognition (ICFHR) IEEE; 2018. p. 587–592.
19. Muangprathub J, Shichim O, Jaroensuk Y, Kajornkasirat S. Automatic grading of scanned multiple choice answer sheets. *International Journal of Engineering and Technology(UAE)* 2018;7:175–179.
20. Zampirolli FDA, Gonzalez JAQ, de Oliveira Neves RP. Automatic Correction of Multiple-Choice Tests using Digital Cameras and Image Processing. In: IX Workshop de Visão Computacional; 2013. p. 1–6.
21. Abbas AA. An Automatic System to Grade Multiple Choice Questions paper based exams. *Journal of university of Anbar for Pure science* 2009;3:174–181.
22. Saengtongsrikamon C, Meesad P, Sodsee S. Scanner-Based Optical Mark Recognition. *Journal of Information Technology* 2009;5:69–73.
23. Gharavi-Alkhansari M. A fast globally optimal algorithm for template matching using low-resolution pruning. *IEEE Transactions on Image Processing* 2001;10:526–533.
24. Pai NR, Kolkure VS. Design and Implementation of Optical Character Recognition Using Template Matching for Multi Fonts/Size. *IJRET: International Journal of Research in Engineering and Technology* 2015;4:2321–7308. <http://www.ijret.org>.
25. Hsiao IH. Mobile grading paper-based programming exams: Automatic semantic partial credit assignment approach. In: *European Conference on Technology Enhanced Learning 2016*, vol. 9891 LNCS Springer; 2016. p. 110–123.
26. Loke SC, Kasmiran KA, Haron SA. A new method of mark detection for software-based optical mark recognition. *PLoS ONE* 2018;13.
27. Fisteus JA, Pardo A, García NF. Grading Multiple Choice Exams with Low-Cost and Portable Computer-Vision Techniques. *Journal of Science Education and Technology* 2013;22:560–571.
28. Jiao J, Wang X, Deng Z, Cao J, Tang W. A fast template matching algorithm based on principal orientation difference. *International Journal of Advanced Robotic Systems* 2018;15:1–9.
29. Čupić M, Brkić K, Hrkać T, Kalafatić Z. Supporting automated grading of formative multiple choice exams by introducing student identifier matrices. In: *34th International Convention on Information and Communication Technology, Electronics and Microelectronics - Proceedings*; 2011. p. 993–998.
30. Loke SC, Kasmiran KA, Haron SA. A Software Application for Survey Form Design and Processing for Scientific Use. *IEEE Access* 2017;5.

How to cite this article: Kommey B., Keelson E., Samuel F., Twum-Asare S., Akuffo K.K. (2022), Automatic Multiple Choice Examination Questions Marking and Grade Generator Software, *IPTEK The Journal of Technology and Science*, 33(3):175-189.