Image Processing & Computer Vision
Project Proposal
Team 6

Team Members

Mohammed Mohammed Saad	1170044
Mariam El Baz	1170546
Moaaz Ashraf Zaki	1170353
Fatema Fawzy Mohamed Sabry	1170530

Table of Contents

Project Idea and Need

There are a lot of possible and cool ideas for this project. However, we decided to go for the grades auto filler referenced in the optional document, as our main focus is to implement an idea that is practical and meets the needs of others rather than just implementing something with no potential use. The basic idea is to implement a system that takes a captured photo of a grades sheet or a bubble sheet that is filled by hand and use it to autofill an excel sheet. The system should deal with different inputs for the photo including different photo angles, use of different ink colors, different sheet formats, different number of students/questions, and so on as mentioned in the project document.

Time is the biggest fortune anyone can have, so there is no need to waste it on trivial tasks like manually copying grades from a paper into an Excel sheet. Moreover, if tasks like filling in the students' grades from a bubble sheet can be made faster, it will spare more time for doing more useful things. For the sake of time saving and automating the grading process for our faculty's staff members, we decided to take on this project. Hopefully, it can also reduce the amount of human errors due to manually copying grades from a hard copy to a soft copy.

Block Diagrams

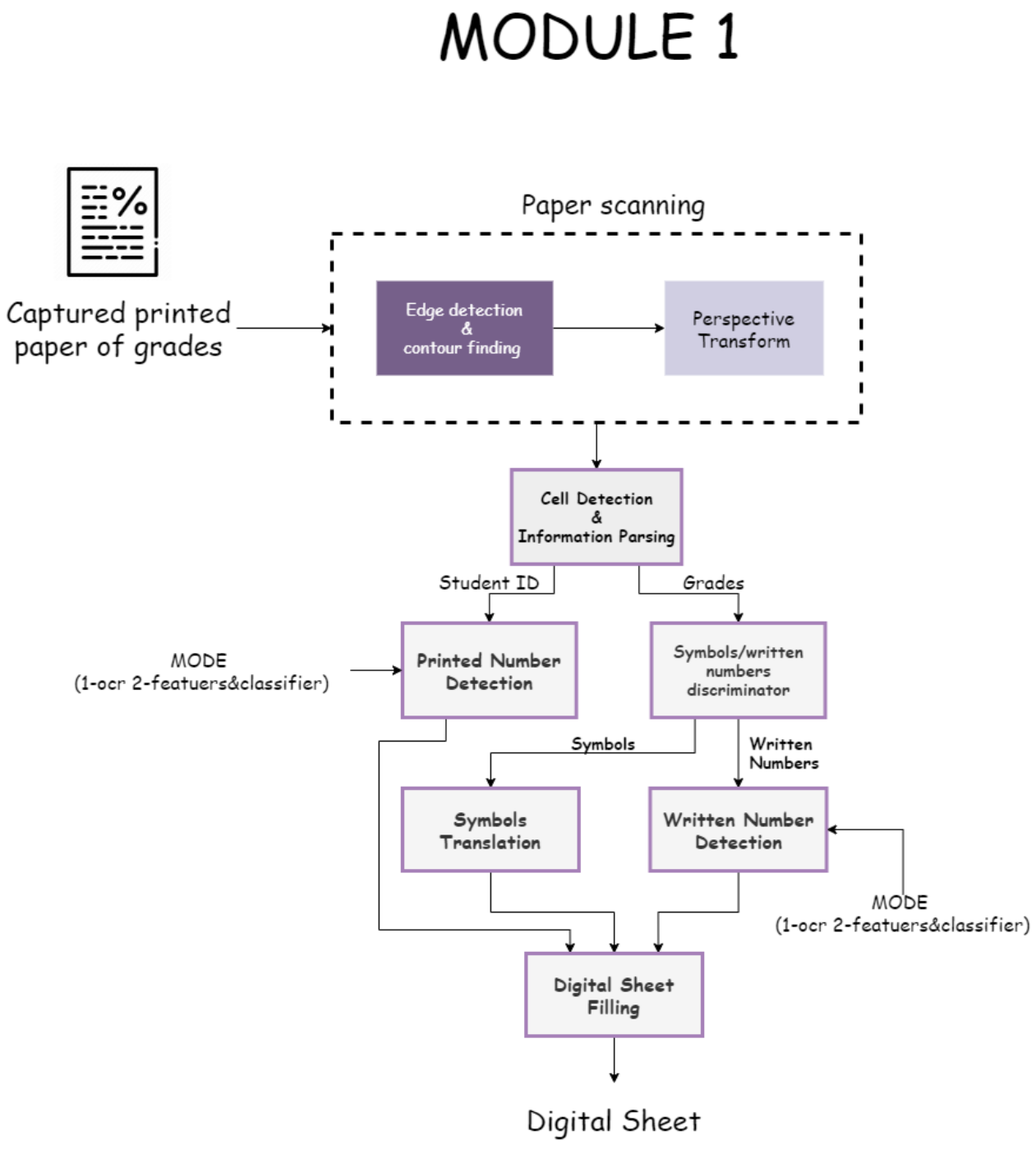


Figure 1: Module 1 Block Diagram

Block	Built-in/ Implemented	Implementation
Edge Detection	**Built-in**	**skimage.feature.canny**(image, sigma=1.0, low_threshold=None, high_threshold=None, mask=None, use_quantiles=False, *, mode='constant', cval=0.0)
Contour Finding	**Built-in**	**cv.find_contours**(image, level=None, fully_connected='low', positive_orientation='low', *, mask=None)
Perspective Transform	**Built-in**	**cv.getPerspectiveTransform**(src, dst[, solveMethod]) -> retval
Cell Detection	**Implemented**	Using kernels and morphological operations
Printed Number Detection	**Built-in**	OCR with Pytesseract
Symbols and written numbers discriminator	**Implemented**	Depending on the column where the data falls
Symbols Translation	✔	Using statistics of angles, lines and ellipses
	□	
	-	
	Stacked Vertical Lines	
	Stacked Horizontal Lines	
	?	
Written Number Detection	**Implemented**	Features + classifier
Digital Sheet Filling	**Built-in**	Using pandas dataframes

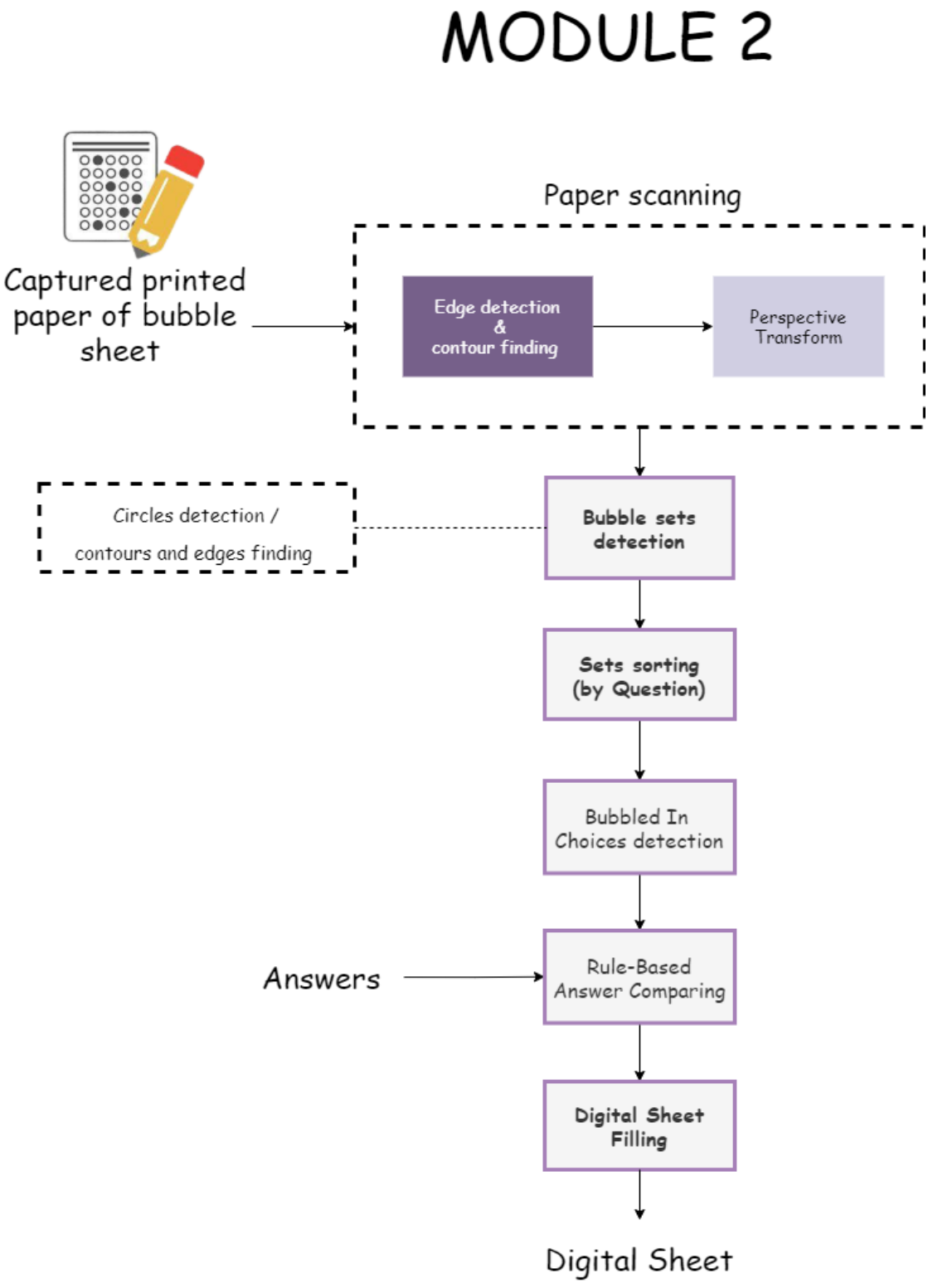


Figure 2: Module 2 Block Diagram

Block	Built-in/ Implemented	Implementation
Edge Detection	**Built-in**	**skimage.feature.canny**(image, sigma=1.0, low_threshold=None, high_threshold=None, mask=None, use_quantiles=False, *, mode='constant', cval=0.0)
Contour Finding	**Built-in**	**skimage.measure.find_contours**(image, level=None, fully_connected='low', positive_orientation='low', *, mask=None)
Perspective Transform	**Built-in**	**cv.getPerspectiveTransform**(src, dst[, solveMethod]) -> retval
Circles Detection	**Built-in**	Using OpenCV hough circles
Bubble Sets Detection	**Implemented**	Bounding box extraction, sorting and arranging.
Sets Sorting		
Bubbled In Choices Detection		Extracting answers with calculating area of filled pixels.
Rule Based Answer Comparison	**Implemented**	By comparing the detected bubbled in choices with the provided model answer
Digital Sheet Filling	**Built-in**	Using pandas dataframes

Datasets

MNIST dataset was used for training the hand-written digits classifier.

For testing:

We printed out papers and filled them out manually.

You can find the full produced dataset through the following links:

1) **Module1**
   https://drive.google.com/file/d/1M8D-dohijVlG2mfxhIuvALO4p8dcc5EP/view?usp=sharing
2) **Module2**
   https://drive.google.com/drive/folders/19pC2qLsh7pIVJNtvxyYzr1bcJg2sor39?usp=sharing

Used Algorithms

Grades Sheet Module

Scanner:

1) Canny edge detector .
2) Closing.
3) Contours finding.
4) Contours approximation to a polygon.
5) Perspective transformation.

Cell detection and information parsing:

1) Vertical and horizontal lines detection
2) Addition (Or.)
3) Erosion, otsu thresholding,
4) Information extraction (using xoring and inversion)
5) Bounding boxes and contours sorting.
6) Removing noisy lines (using abnormal values filtering)
7) Arranging bounding boxes into sorted rows and columns from up to bottom and left to right.
8) Detecting the potentially wrong rows using the mod of the number of columns per each row.
9) Correcting noisy rows using the statistically approximate widths for each column.

Hand-written digits detection:

1) Thresholding with constant value.
2) Histogram of gradients.
3) Black/White transition areas.
4) Curvature transformation features.
5) ~~Hough lines~~. (not used)
6) ~~Hough ellipses.~~ (not used)
7) K-nearest neighbour
8) ~~SVM~~ (not used)

Note: A pretrained model was used but performed poorly, so it was removed.

Printed digits detection:

1) Using adaptive gaussian thresholding.
2) Upsizing the digits using cubic interpolation.

3) Applying OCR using Pytesseract.

Note: the same features for hand-written digits were tested here as well but failed.

Symbols descrimination:

1) Using adaptive gaussian thresholding.
2) Upsizing the digits using cubic interpolation.
3) Bounding box extraction.
4) Hough lines transformation.
5) Hough ellipses transformation.

Bubble Sheet Module

Scanner:

1) Canny edge detector.
2) Closing.
3) Contours finding.
4) Contours approximation to a polygon.
5) Perspective transformation.

Preprocessing steps:

1) Canny edge detector.
2) Closing.

Circles Detection:

1) Hough circles transform.

Circles to arranged bounding boxes:

1) Sorting circles from top to bottom.
2) Extracting sorted rows from top to bottom and left to right.
3) Correcting potentially wrong rows using
    a) Statistically start and end of each row.
    b) Intersection over union to remove overlapping bounding boxes.

Extracting information from bounding boxes:

1) Area of filled pixels.

How to Use

The code:

1) Unzip the project folder.
2) cd to server
3) pip3 install -r requirements.txt (or you can use docker to install and run through the commands in server/docker_script.txt)
4) Run each file individually if you wish (python filename.py)
5) Or run the whole app using python app.py or flask run
6) You can also find some test .ipynb files in the playground folder if you want to test something.
7) Some functions have a parameter `visualize`, set it to true if you want to see the photos being processed step by step.
8) If you wish, you can also run the client locally by cd client then npm start.

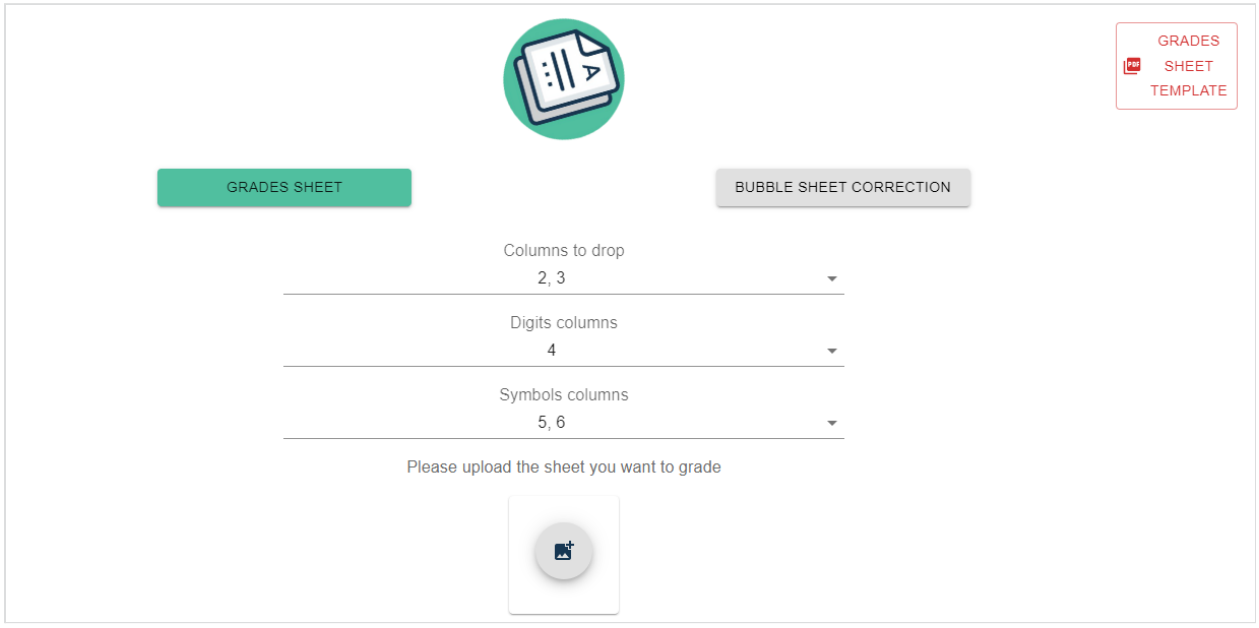
The website:

- Server: https://grade-auto-filler.herokuapp.com/
- Client: https://sheetgrader.web.app/

  Notes:

  1) Use localhost for better experience
  2) Some inputs may not processed because of heroku restrictions on response time
  3) You have to get the link of downloading output file from inspect -> network -> preview because heroku restrict auto download of .xlsx files

Grades Sheet Moule



How to experiment with the grades sheet module:

1) Download the template at the upper right corner of the page.
2) Edit in the template according to your class list.
3) Print the template.
4) Fill in the grades of your students.
5) Return back to the page.
6) State if there are any columns you need to drop while processing.
7) State which columns include handwritten digits.
8) State which columns include symbols.
9) Take a picture of your grades sheet after filling the desired columns and upload it.
10) Wait a couple of seconds.
11) Download the digitized excel sheet.

Template Example:

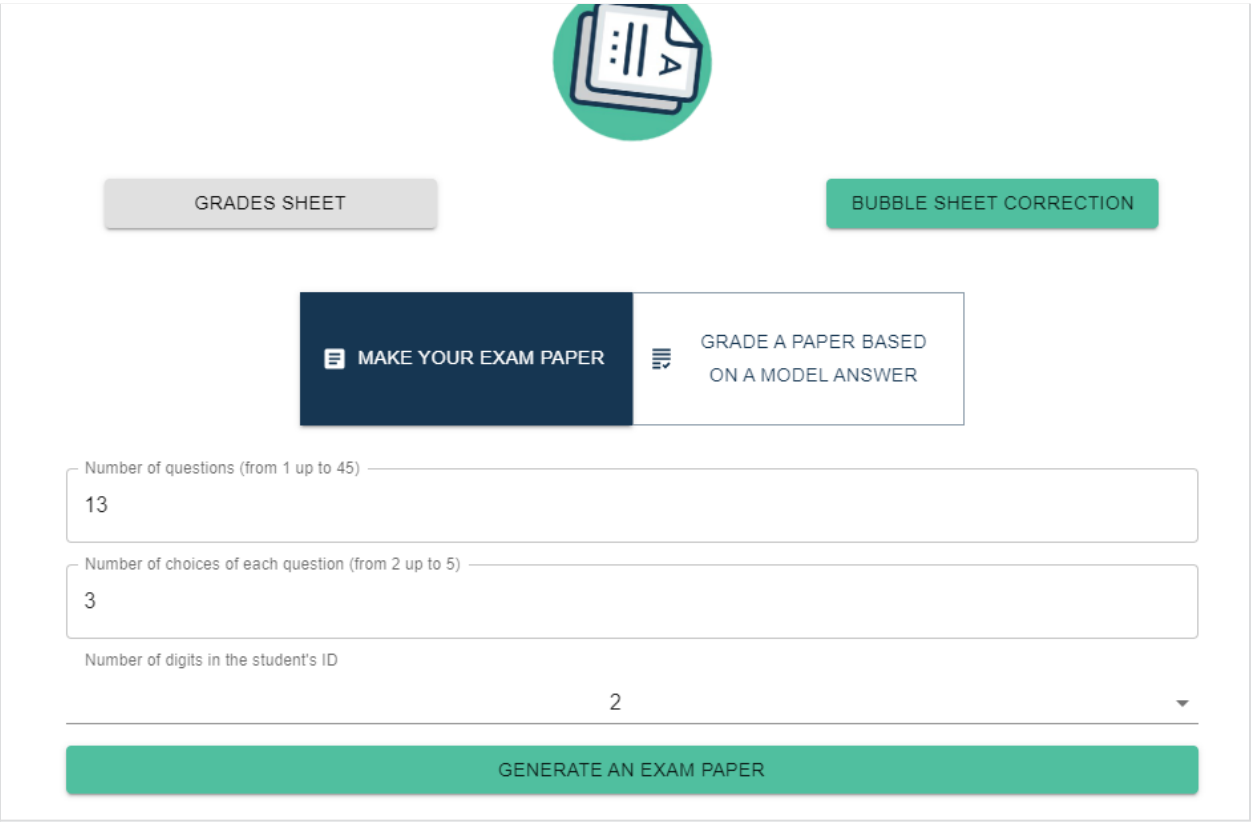
Fall-2021 Class list for Lecture Marketing (GENN326) Location : 20105 - 45 - الجيزة الرئيسي - odd Time : Tuesday(11:13)

Code	Student Name	English Name	1	2	3
1180236	احمد معتز لطفى احمد	Ahmed Motaaz Lotfy Ahmed			
1180333	حبيبة عصام حسب الله توفيق عمران	Habiba Essam Hassaballah Tawfik Omran			
1180128	سعد الدين محمد سعد محمد	Saad El-din Mohamed Saad Mohamed			
1180255	عبد الله محمد جلال السحيمى	AbdAllah Mohammed Galal El-Suhaimi			
1180274	علا ايمن عبدالفتاح المغربى	Ola ayman abdelftah elmaghraby			
1180056	على شريف على حسب الله	Ali Sharif Ali Hasb Allah			
1180041	عمر محمد فتحى شلقامى شعراوى	Omar Mohamed Fathy Shalkkamy Shaarawy			
1180606	فاطمة عصام محمد جاب الله	Fatma Issam Mohamed Gaballah			
1180456	فرح اسامه زين الدين محمد	farah ossama zein elden			

Sample of the generated excel file:

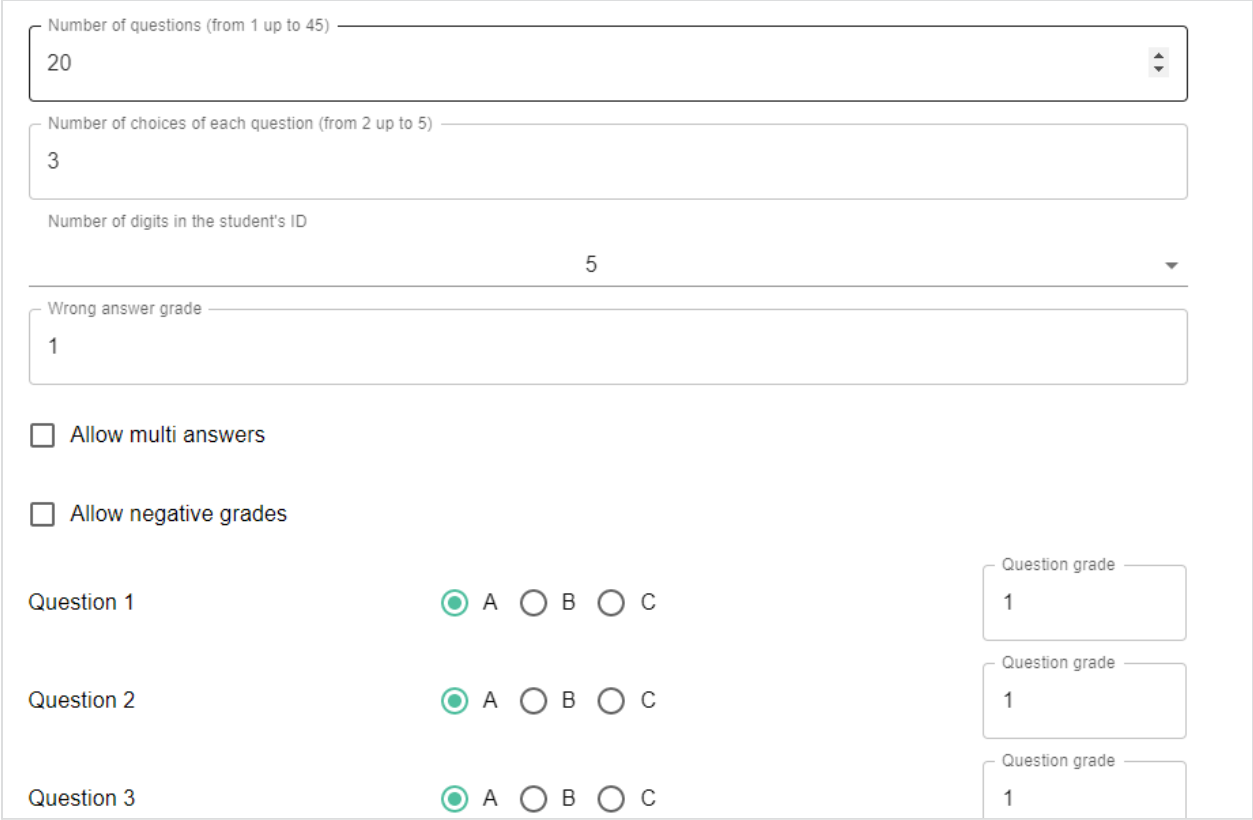
	A	B	C	D	E	F	G
1		id	1	2	3	4	5
2	0	1960236			9	5	0
3	1	1180333			4	3	0
4	2	1180128			7	5	5
5	3	1160255			3		2
6	4	1130274			1		4
7	5	1180056			4	0	0
8	6	1380041			6	3	5
9	7	1380606			4		5
10	8	1380456			4	0	
11	9				4	5	0
12	10	1160582			9		
13	11	1160207			8	3	5
14	12	1130045			2	5	5
15	13	1130212			4		2
16	14	80155			4	5	
17	15	1170343			9	0	
18	16	1180172			4		5

Bubble Sheet Module



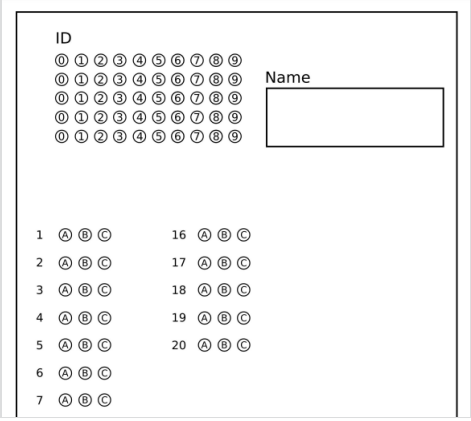
How to use the bubble sheet module

1) Choose the number of questions you want.
2) Choose the number of choices in each question.
3) Determine how many digits are used for each student ID.
4) Generate the exam paper.
5) Download the generated paper.
6) Print as many copies as you want for your students.
7) Carry out your exam.



8) Come back to the page and visit the other tab.
9) Determine the wrong answer grade.
10) Select whether you allow multi answers or not.
11) Select whether you allow negative grades or not.
12) For each question choose the model answer/answers for that question and give it a weight.
13) Upload the papers you collected from the students after the exam.
14) Wait for a couple of seconds.
15) Download an excel sheet containing the students' grades.

Sample of the generated bubble sheet and output of grader:

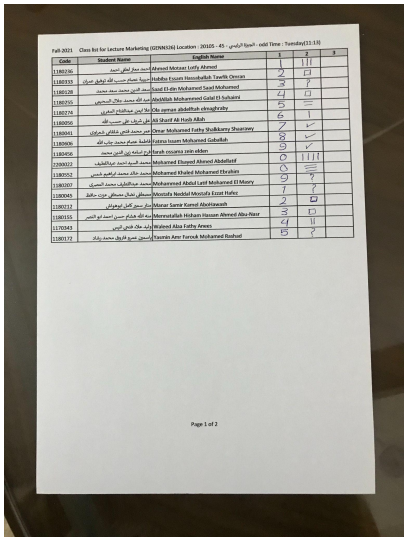


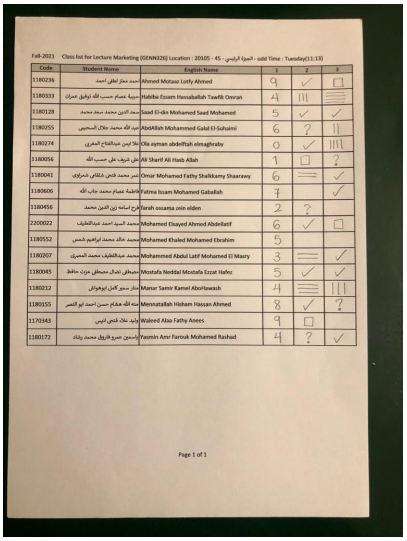
	Q(1)	Q(2)	Q(3)	Q(4)	Q(5)
1170044	1	-1	-1	-1	-1
1998876	1	-1	-1	-1	-1

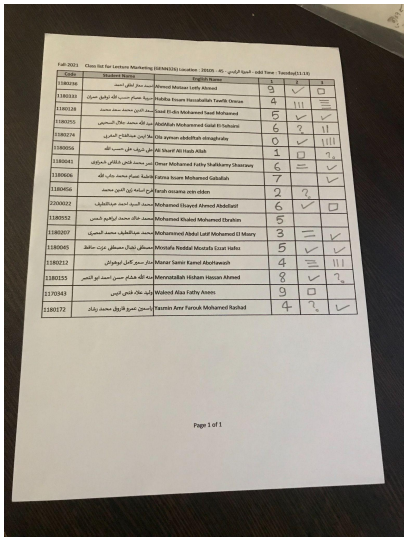
Experiment Results & System Analysis

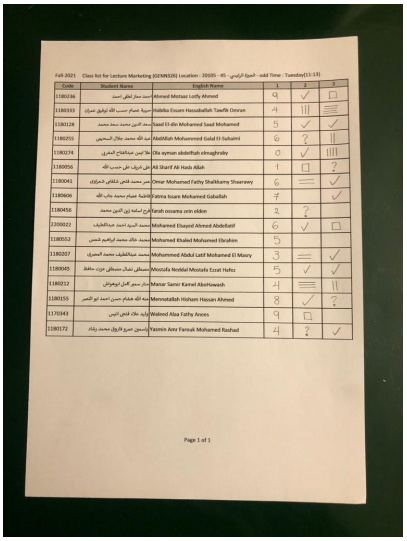
Grades Sheet Module

To conduct a fair experiment that shows system strengths and weaknesses, 15 sheet papers with different cameras and environments are used, the following are some samples of sheets:









Responses time (on web application):

- **Grade sheet:** 35 sec +/- 2sec

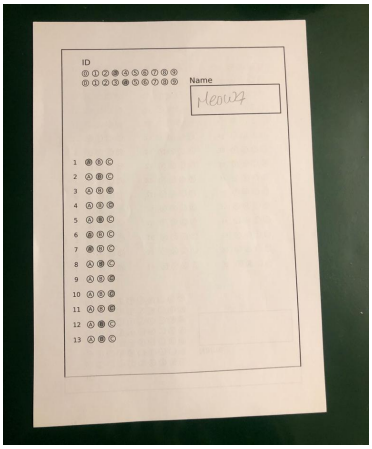
Bubble Sheet Module

To conduct a fair experiment that shows system strengths and weaknesses, many exam papers with different cameras and environments are used with the following types:

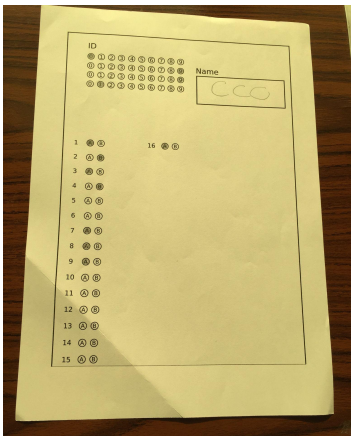
1) 15 papers with 2 ID digits, 13 questions and 3 choices per question
2) 8 papers with 4 ID digits, 16 questions and 2 choices per question
3) 2 papers with 4 ID digits, 40 questions and 5 choices per question
4) 13 papers with 7 ID digits, 45 questions and 4 choices per question

Sample of test images:

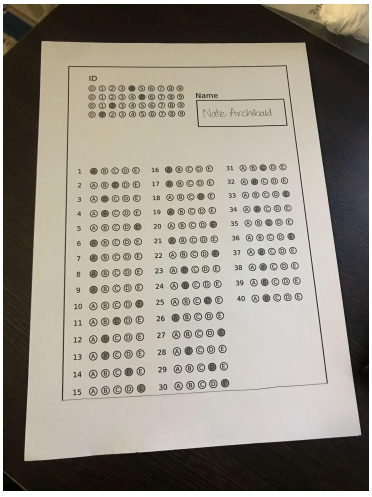
Type 1



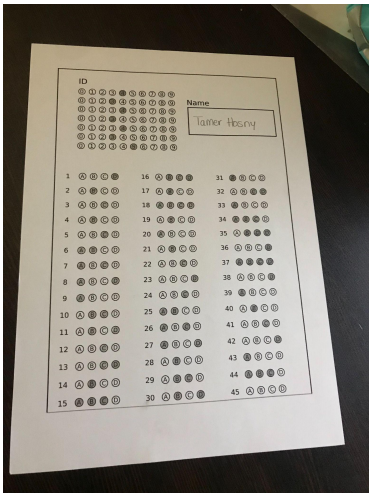
Type 2



Type 3



Type 4



Responses time (on web application):

- **Page generation:** 435ms +/- 15ms
- **Page grading:** 1.5s +/- 0.75s

Accuracy & Performance

Grade Sheet Module

Student ID:

Paper	Classified correctly	Classified wrongly	Accuracy
1	13	4	76.47%
2	5	12	29.41%
3	15	2	88.23%
4	11	6	64.71%
5	10	7	58.8%
6	9	8	52.94%
7	9	8	52.94%
8	11	6	64.71%
9	11	6	64.71%
10	10	7	58.82%
11	10	7	58.82%
12	11	6	67.7%
13	11	6	67.7%
14	6	9	35.29%
15	12	5	70.59%

Handwritten digits:

Training accuracy on MNIST: **91.5%**

From the whole dataset (15 pages) the following statistics are calculated for each digit:

Digit	Classified correctly	Classified wrongly	Accuracy
0	9	12	42.85%
1	11	5	68.75%
2	6	18	25.00%
3	17	8	68.00%
4	17	14	54.83%
5	6	25	19.35%
6	23	7	76.67%
7	8	9	47.06%
8	6	11	35.30%
9	19	8	70.37%

Symbols:

From the whole dataset (15 pages) the following statistics are calculated for each digit:

Symbol	Classified correctly	Classified wrongly	Accuracy
Vertical lines	57	12	82.60%
Horizontal lines	61	4	93.84%
Ticks	87	37	70.14%
Squares	106	14	88.33%
Question marks	11	87	11.22%
Blank	32	0	100%

Bubble Sheet Module

Type 1:

ID: 2 Digits
Number of Questions: 13
Number of Choices: 3

Paper	Classified correctly	Classified wrongly	Accuracy
Meow	13	2	86.67%
Meow4	15	0	100%
Meow11	14	1	93.3%
Meow8	10	5	66.67%
Meow9	12	3	80%
Meow1	14	1	93.3%
Meow12	13	2	86.67%
Meow10	15	0	100%
Mariam El Baz	15	0	100%
Mohammed Saad	15	0	100%
Meow6	15	0	100%
Meow7	15	0	100%
Meow2	15	0	100%
Meow5	15	0	100%
Meow3	13	2	86.67%

Average Accuracy = 92.88%

Type 2:

ID: 4 Digits
Number of Questions: 16
Number of Choices: 2

Paper	Classified correctly	Classified wrongly	Accuracy
Cat P	19	1	95%
Zeow	20	0	100%
Keow	20	0	100%
Neow	19	1	95%
Ceow	20	0	100%
Meow	20	0	100%
CCC	19	1	95%
Wrong ID	14	6	70%

Average Accuracy = 94.37%

Type 3:

ID: 4 Digits
Number of Questions: 40
Number of Choices: 5

Paper	Classified correctly	Classified wrongly	Accuracy
Nate Archilbald	44	0	100%
Moaaz Saad	44	0	100%

Average Accuracy = 100%

Type 4:

ID: 7 Digits
Number of Questions: 45
Number of Choices: 4

Paper	Classified correctly	Classified wrongly	Accuracy
Nicki Minaj (One answer)	52	0	100%
Nicki Minaj (multi answer)	52	0	100%
Mohammed	51	1	98.07%
Meow	44	8	84.61%
Meow	52	0	100%
Meow1	52	0	100%
Tamer Hosny	45	7	86.54%
Tamer Hosny	44	8	84.61%
Meow2	52	0	100%
Meow2	50	2	96.15%
Jeow	52	0	100%
Meow4	52	0	100%
Meow3	51	1	98.07%
Ahmed El Baz	45	7	86.54%

Average Accuracy = 95%

Conclusion

Image processing is one of the most powerful tools that we can use in order to handle our every-day tasks. However, it could become very challenging to find a good model that fits all scenarios in real life as there are countless possibilities in terms of orientation, shadow, quality…etc. In this project we try to limit the user to certain guidelines to follow to facilitate this process and make it compatible with our model. Hence, we provide a template for the sheets module, and make it possible for the users to generate their own bubble sheet according to their desired parameters in terms of number of questions , number of choices..etc. but the generated paper would follow some guidelines that the model can understand and deal with and at the same time is user friendly and not different from the common known bubble sheets.

Additional Comments

1) The accuracy of the images taken with high quality cameras is almost 100% in all bubble sheet papers.
2) The OCR accuracy of the printed digits is calculated on the whole ID level not on single digits. The accuracy would have been better if it was calculated for single digits as most IDs were wrong because it classified only one digit wrongly. But we used the criteria of the whole ID as it is more realistic.
3) The model performs better on some symbols than others, but its overall accuracy is acceptable.
4) The hand-written digits accuracy is guaranteed to be increased if we were allowed to use deep learning.
5) There is no good OCR for hand-written digits, so features+classifier method is the only used one.
6) Performance of hand-written digits model on printed is poor, and there is no available dataset for printed digits that meets the need of the project, so OCR is used only.

Work Division

Name	Tasks
Mohammed Saad	Cell detection - Symbols/numbers discriminator - Handwritten and printed numbers detection (ocr)
Mariam El Baz	Circle detection - Sets sorting - Digital sheet filling
Moaaz Ashraf	Paper scanning - Symbol Translation - Handwritten and printed numbers detection (features classifier)
Fatema Fawzy	Rule based Answer comparison - Bubble in sets detection - Bubbled In Choices

References

Alomran, M., & Chai, D. (2018, August). Automated Scoring System for Multiple Choice Test with Quick Feedback. *International Journal of Information and Education Technology*, *8*. http://www.ijiet.org/vol8/1096-JR296.pdf

Bahara, V., Ranjan, A., & RezaHigh, M. (n.d.). An Innovative Image-based Tabular DataExtraction Parallel Algorithm. *High Performance Computing Lab,National Institute of Science and Technology,Berhampur 761008, India*. https://www.researchgate.net/publication/340214505_An_Innovative_Image-Based_Tabular_Data_Extraction_Parallel_Algorithm

Ionete, B., & Lambrescu, I. (n.d.). Automatic Evaluation of Scanned Multiple Choice Tests. *BULETINUL*, *LXII*, 66 - 71. http://www.unde.ro/bmif/docs/20102/pdf_final_9%20Ionete.pdf

Konya, I., Eickeler, S., & Seibert, C. (2010). Fast Seamless Skew and Orientation Detection in Document Images. *Fraunhofer Institute for Intelligent Analysis and Information Systems (IAIS)Schloss Birlinghoven, 53754 Sankt Augustin, Germany*. https://www.researchgate.net/publication/220930964_Fast_Seamless_Skew_and_Orientation_Detection_in_Document_Images