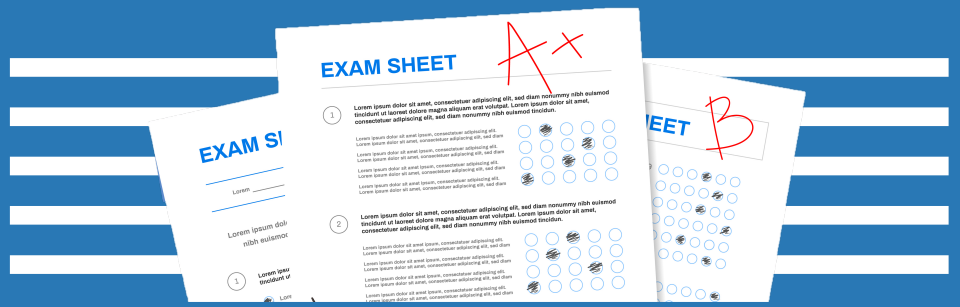
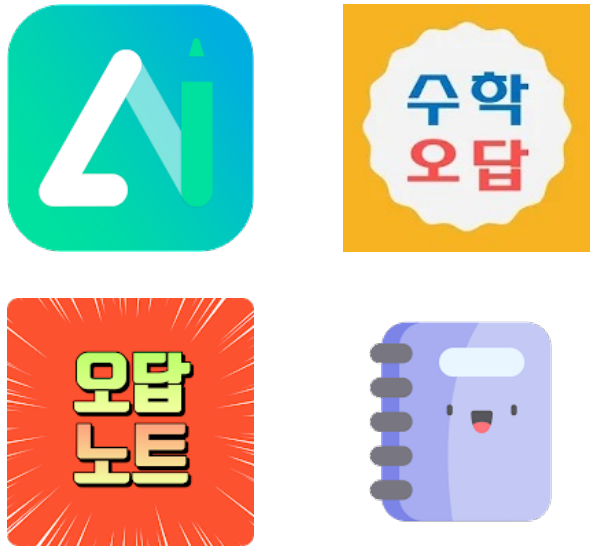
Capstone Design Project SWE3028_41



Final Presentation

TEAM FANCY

Project Objective/Motivation



Current review notes just function as a simple storage!

Auto grading

Problem regenerating

Functions to encourage students!

Capstone Design Project

Project Progress

성균관대학교 SUNG KYUN KWAN UNIVERSITY(SKKU)

Minji Daehee Eunji Daeyeol

WEEK	5	6	7	8	9	10	11	12	13	14	15
	9/27~10/3	10/4~10/10	10/11~10/17	1018~10/24	10/25~10/31	11/1~11/7	11/8~11/14	11/15~11/21	11/22~11/28	11/29~12/05	12/06~12/10
EVENT						***Midterm Presentation***					***Final Presentation***
Develop idea											
- Design feature for motivation	**ALL**										
- Study core technology	**ALL**										
FRONT / BACK											
- Build server with Flask		Minji									
- DB setting		Minji									
- UI/UX/flow design		Daeyeol / Eunji	Daeyeol / Eunji	Daeyeol / Eunji	Daeyeol / Eunji						
- Frontend implementation		Eunji	Eunji	Eunji	Eunji	Eunji	Eunji	Eunji	Eunji	Eunji	
- App api server implementation			Daeyeol / Minji	Daeyeol	Daeyeol	Daeyeol	Daeyeol	Daeyeol	Daeyeol	Daeyeol	
- Review note generation							Eunji / Daeyeol	Eunji / Daeyeol	Eunji / Daeyeol	Eunji / Daeyeol	
MODEL											
- Image dewraping		Daehee									
- Collect data for object detection(Question)	Daehee/Minji	Daehee/Minji	Daehee/Minji								
- Collect data for object detection(answer)		Daehee/Minji	Daehee/Minji								
- Labeling object area				ALL	ALL						
- Build YOLO model and train				Daehee/Minji	Daehee	Daehee					
- Run YOLO in web server							Daehee	Daehee			
- Implement YOLO related API functions									Minji	Minji	
- Detect user's answer									Daehee	Daehee	
- Auto Grading									Minji/Daehee	Minji/Daehee	
NEXT STEP											
- Regenerate question feature					Minji	Minji	Minji	Minji			
- Add feature for motivation							Eunji / Daeyeol	Eunji / Daeyeol	Eunji / Daeyeol	Eunji / Daeyeol	

ROLES

Minji

- Web server / DB Modeling
- Save / Load Image
- YOLO related API functions
- Data Labeling

Daehee

- Train YOLO
- Run YOLO in Web Server
- Process YOLO output
- Data Labeling

Daeyeol

- UI/UX design
- DB Modeling
- Implement API functions
- Data Labeling

Eunji

- UI/UX design
- Implement Application Features
- Data Labeling

System Architecture

Client

Flutter

Google's UI toolkit
for building applications

API Server

VULTR

Flask
web development,
one drop at a time

Python web framework

PostgreSQL

AI

docker

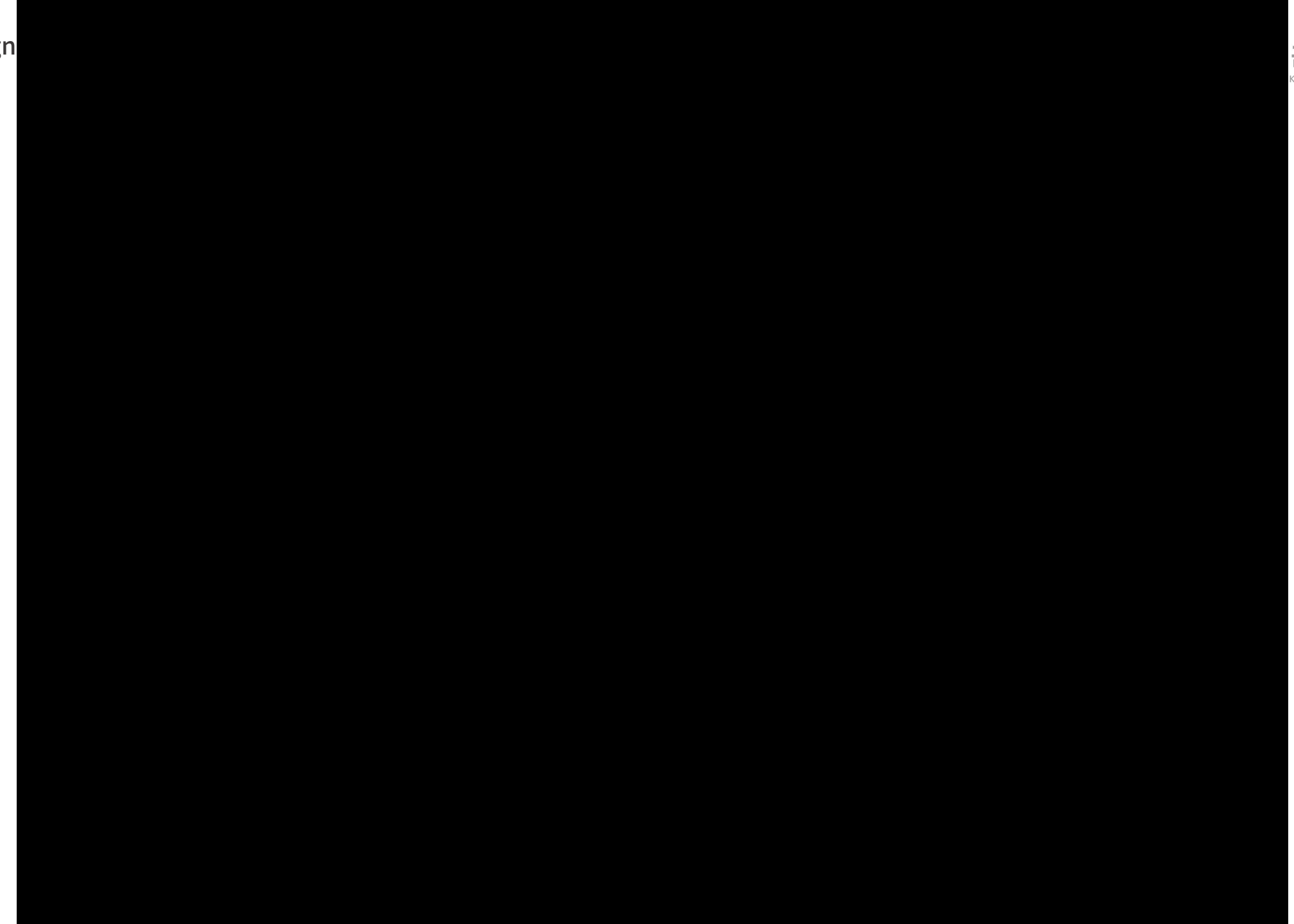
YOLO

Storage

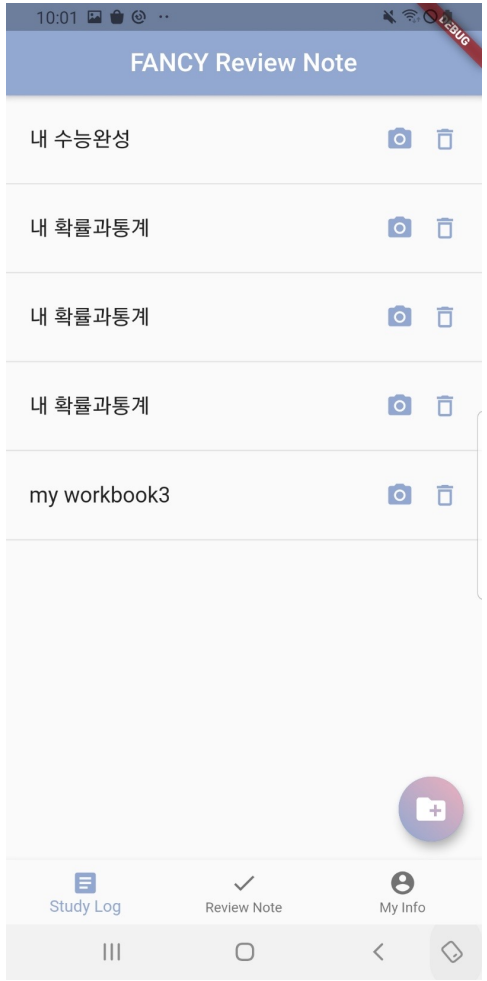
Azure Blob Storage

Limitations before watching demo video…

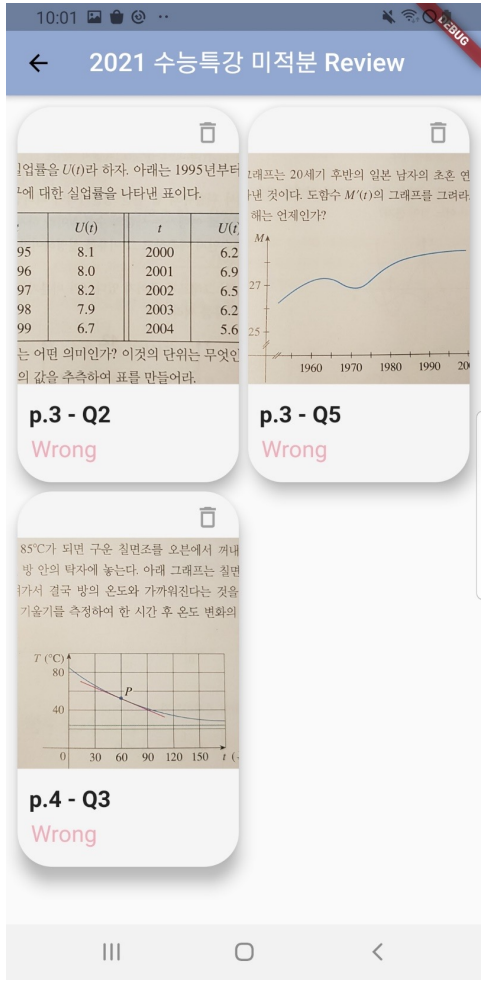
- Only grade multiple choice questions
- User should mark the answer like as marking on OMR
- Recommend to use 수능특강/수능완성 for auto-grading function.
  - Might not working on workbooks with a format not similar with 수능특강/수능완성.



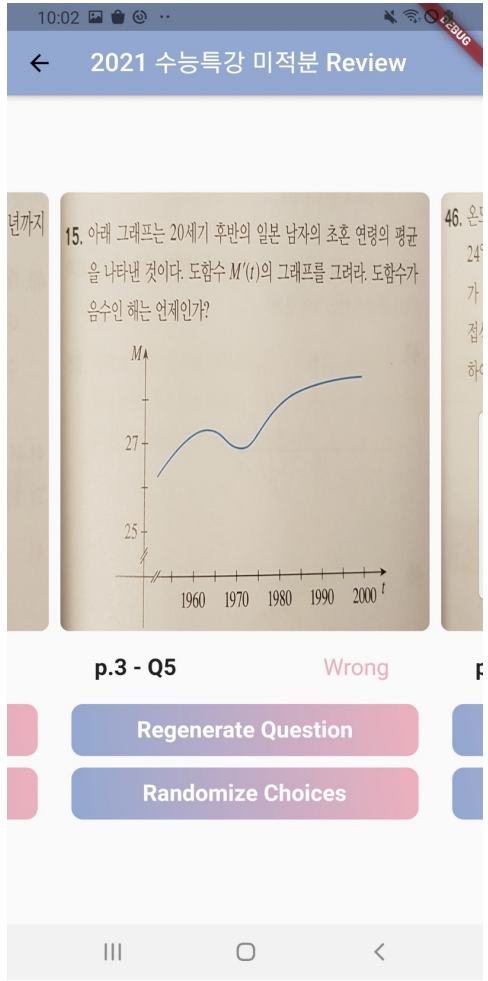
Final Design - UI/UX



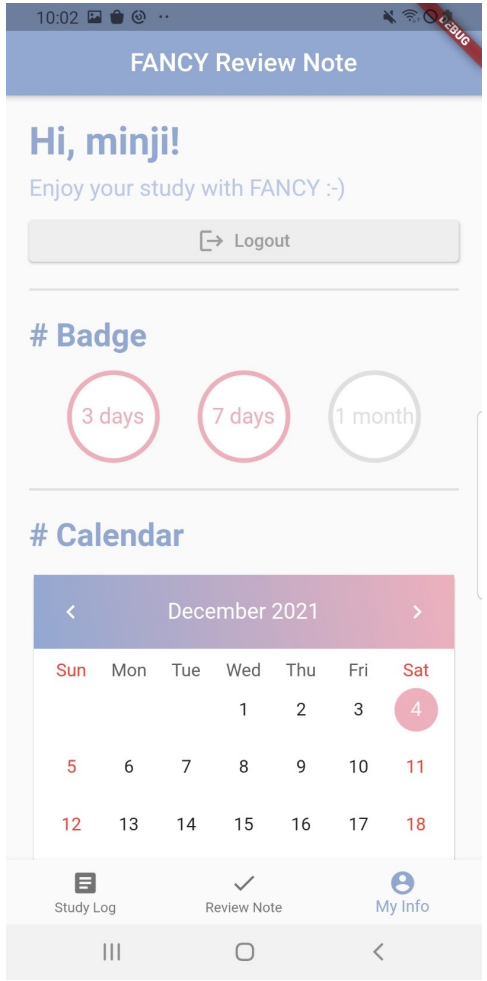
Review note list page



Problem list page



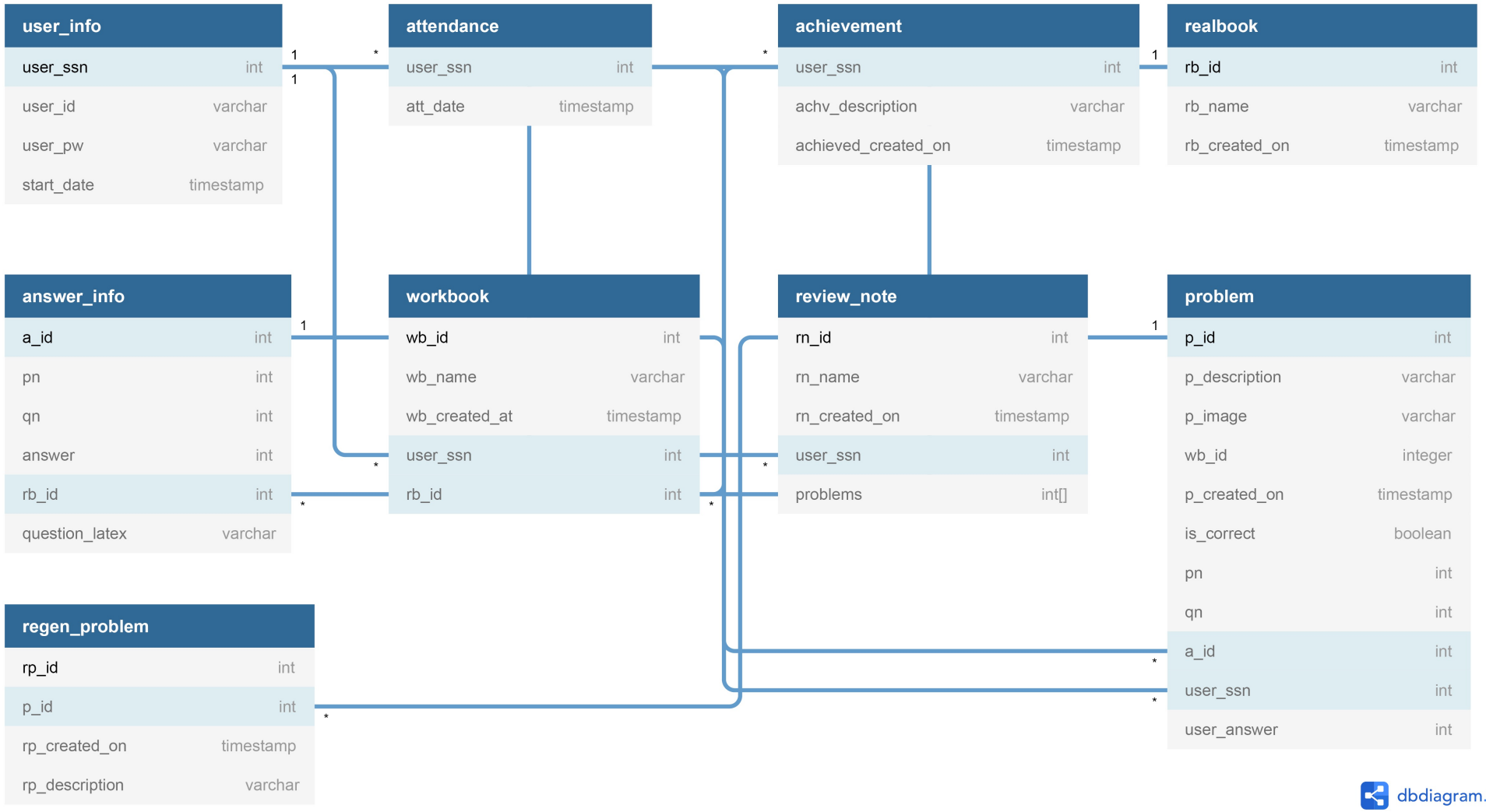
Individual problem page



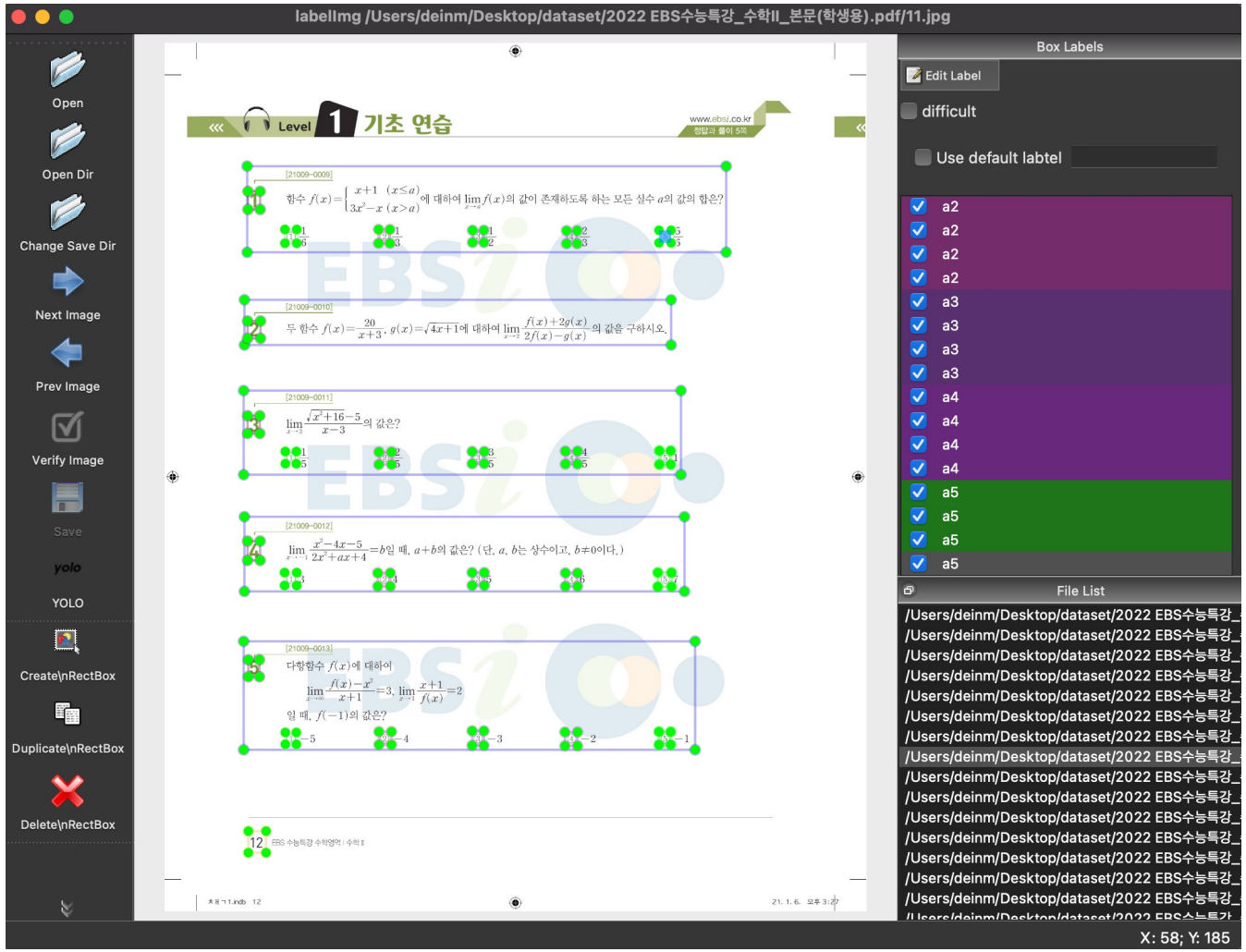
User info page

Capstone Design Project

Final Design - Database



AI - Dataset



Total 8 books, 400 pages, 1000 questions

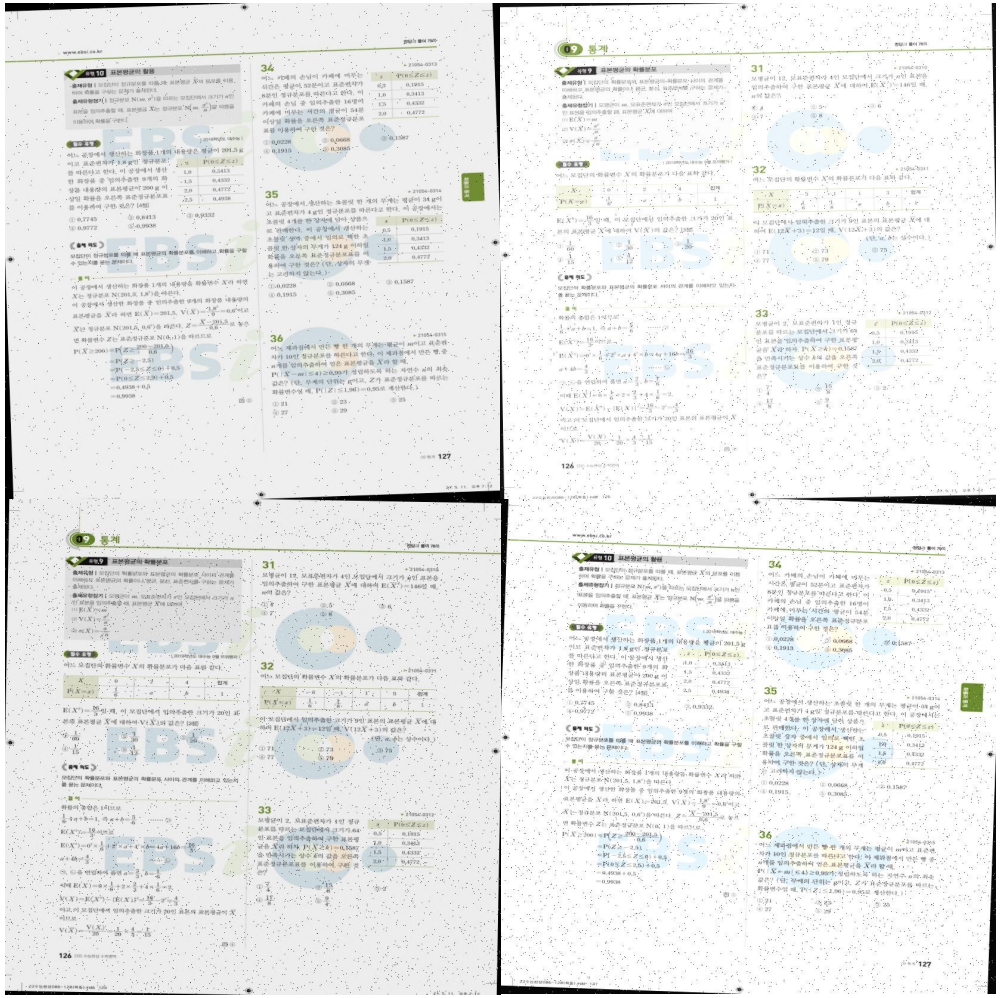
- q(question)
- qn(question number)
- a1~a5(answer1~5)
- pn(page number)

AI - Dataset

TRAIN / TEST SPLIT

Training Set	Validation Set	Testing Set
92%	5%	3%
906 images	**51** images	**26** images

PREPROCESSING	**Resize:** Stretch to 832×832
AUGMENTATIONS	**Outputs per training example:** 3 **Rotation:** Between -2° and +2° **Brightness:** Between -7% and +7% **Exposure:** Between -8% and +8% **Noise:** Up to 1% of pixels



AI - Model description

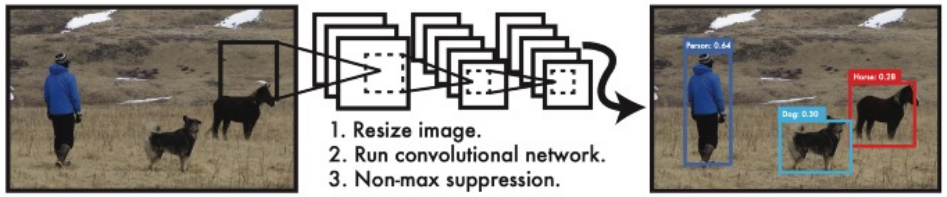
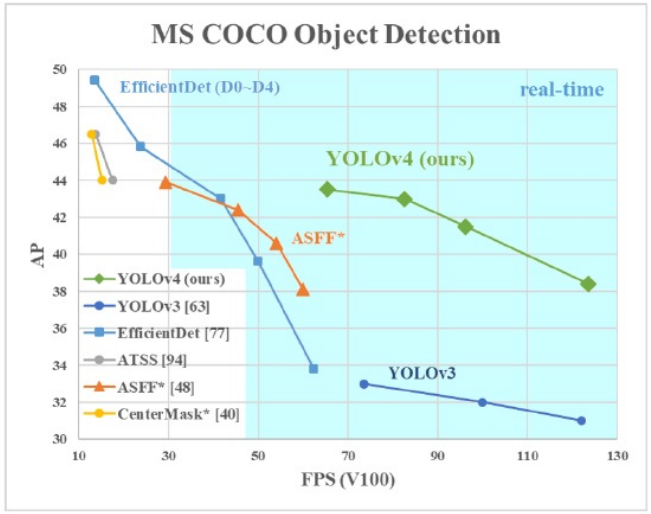


Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448 × 448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.



Should detect all meaningful features to auto grade the image 💯

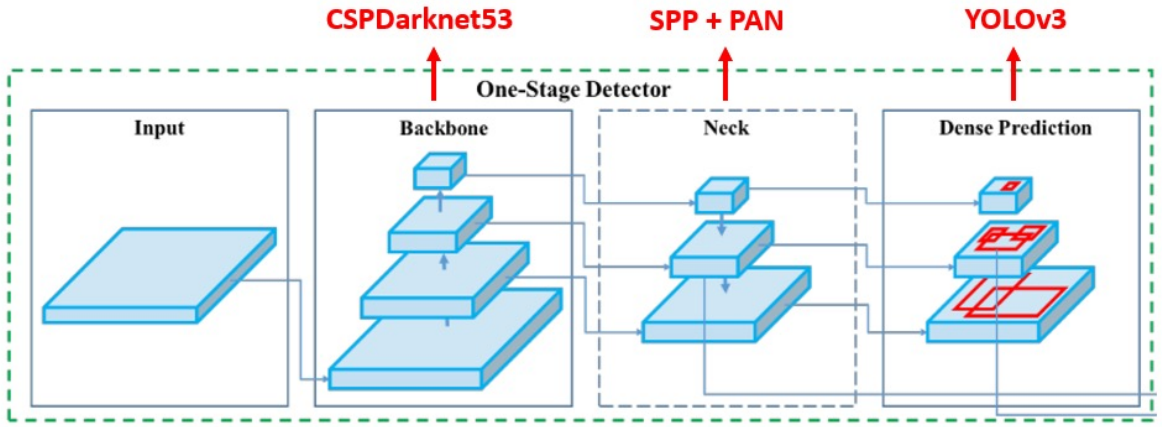
We considered all those features: problem number, answer options, ⋯ as a "object"

YOLO is very fast 🚀

Since it detects object with an algorithm with grid, it is faster than other models

AI - Model description

•**Input** :
416x416 sized image



•**Output** :
52x52, 26x26, 13x13 sized feature maps

YOLOv4 = YOLOv3 + CSPDarknet53 + SPP + PAN + BoF + BoS

↓

Path Aggregation Network

What is YOLOv4?

Basically, YOLOv4 is YOLOv3 with various latest techniques added.
YOLOv4 created the **best performance combination** by combining various backbone, head, and neck.

AI – Input and Output


```
"frame_id":2,
"filename":"data/test/16-marked.jpg",
"objects": [
 {"class_id":7, "name":"qn", "relative_coordinates":{"center_x":0.088511, "center_y":0.184436, "width":0.016257, "height":0.019415}, "confidence":0.984140},
 {"class_id":7, "name":"qn", "relative_coordinates":{"center_x":0.089032, "center_y":0.504020, "width":0.020627, "height":0.018493}, "confidence":0.982832},
 {"class_id":7, "name":"qn", "relative_coordinates":{"center_x":0.088125, "center_y":0.683966, "width":0.022226, "height":0.019109}, "confidence":0.980129},
 {"class_id":7, "name":"qn", "relative_coordinates":{"center_x":0.089482, "center_y":0.344396, "width":0.019841, "height":0.020271}, "confidence":0.950412},
 {"class_id":6, "name":"q", "relative_coordinates":{"center_x":0.483181, "center_y":0.220037, "width":0.919415, "height":0.147278}, "confidence":0.999480},
 {"class_id":6, "name":"q", "relative_coordinates":{"center_x":0.473475, "center_y":0.743913, "width":0.832438, "height":0.178773}, "confidence":0.998936},
 {"class_id":6, "name":"q", "relative_coordinates":{"center_x":0.480875, "center_y":0.378378, "width":0.917356, "height":0.136682}, "confidence":0.998782},
 {"class_id":6, "name":"q", "relative_coordinates":{"center_x":0.479361, "center_y":0.545566, "width":0.825833, "height":0.153383}, "confidence":0.995890},
 {"class_id":5, "name":"pn", "relative_coordinates":{"center_x":0.087907, "center_y":0.905967, "width":0.028873, "height":0.017941}, "confidence":0.936690},
 {"class_id":4, "name":"a5", "relative_coordinates":{"center_x":0.318163, "center_y":0.430582, "width":0.021244, "height":0.014300}, "confidence":0.983725},
 {"class_id":4, "name":"a5", "relative_coordinates":{"center_x":0.333444, "center_y":0.813176, "width":0.021405, "height":0.015173}, "confidence":0.973858},
 {"class_id":4, "name":"a5", "relative_coordinates":{"center_x":0.329343, "center_y":0.571078, "width":0.017106, "height":0.015231}, "confidence":0.964385},
 {"class_id":4, "name":"a5", "relative_coordinates":{"center_x":0.343615, "center_y":0.273188, "width":0.019345, "height":0.014749}, "confidence":0.957099},
 {"class_id":3, "name":"a4", "relative_coordinates":{"center_x":0.147253, "center_y":0.272463, "width":0.019646, "height":0.014591}, "confidence":0.972564},
 {"class_id":3, "name":"a4", "relative_coordinates":{"center_x":0.147105, "center_y":0.431366, "width":0.019852, "height":0.013975}, "confidence":0.881207},
 {"class_id":3, "name":"a4", "relative_coordinates":{"center_x":0.145730, "center_y":0.572350, "width":0.018171, "height":0.014369}, "confidence":0.824575},
 {"class_id":3, "name":"a4", "relative_coordinates":{"center_x":0.142539, "center_y":0.813598, "width":0.020920, "height":0.014761}, "confidence":0.521491},
 {"class_id":2, "name":"a3", "relative_coordinates":{"center_x":0.510761, "center_y":0.550358, "width":0.019185, "height":0.014818}, "confidence":0.958232},
 {"class_id":2, "name":"a3", "relative_coordinates":{"center_x":0.489552, "center_y":0.412049, "width":0.020466, "height":0.012769}, "confidence":0.928133},
 {"class_id":1, "name":"a2", "relative_coordinates":{"center_x":0.317887, "center_y":0.410908, "width":0.020438, "height":0.012071}, "confidence":0.989291},
 {"class_id":1, "name":"a2", "relative_coordinates":{"center_x":0.329175, "center_y":0.549998, "width":0.017544, "height":0.013789}, "confidence":0.981069},
 {"class_id":1, "name":"a2", "relative_coordinates":{"center_x":0.333100, "center_y":0.792784, "width":0.022226, "height":0.013306}, "confidence":0.977361},
 {"class_id":1, "name":"a2", "relative_coordinates":{"center_x":0.343630, "center_y":0.253631, "width":0.018945, "height":0.012571}, "confidence":0.956280},
 {"class_id":0, "name":"a1", "relative_coordinates":{"center_x":0.147870, "center_y":0.251824, "width":0.018447, "height":0.012919}, "confidence":0.969067},
 {"class_id":0, "name":"a1", "relative_coordinates":{"center_x":0.146556, "center_y":0.549848, "width":0.017012, "height":0.014766}, "confidence":0.955346},
 {"class_id":0, "name":"a1", "relative_coordinates":{"center_x":0.147713, "center_y":0.411763, "width":0.017547, "height":0.013414}, "confidence":0.938136},
 {"class_id":0, "name":"a1", "relative_coordinates":{"center_x":0.142855, "center_y":0.792399, "width":0.020920, "height":0.014034}, "confidence":0.798333}
]
}
]
```

AI – Hyperparameters

CFG Parameters in the [net] section

Alexey edited this page on 16 Jun 2020 · 5 revisions

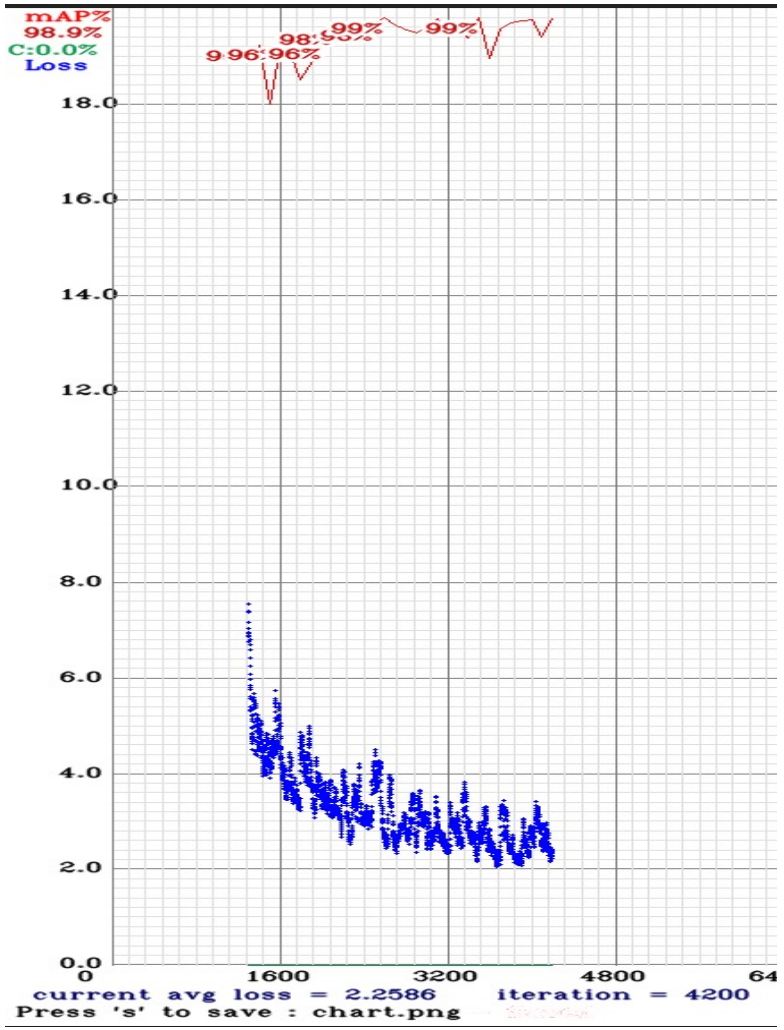
CFG-Parameters in the `[net]` section:

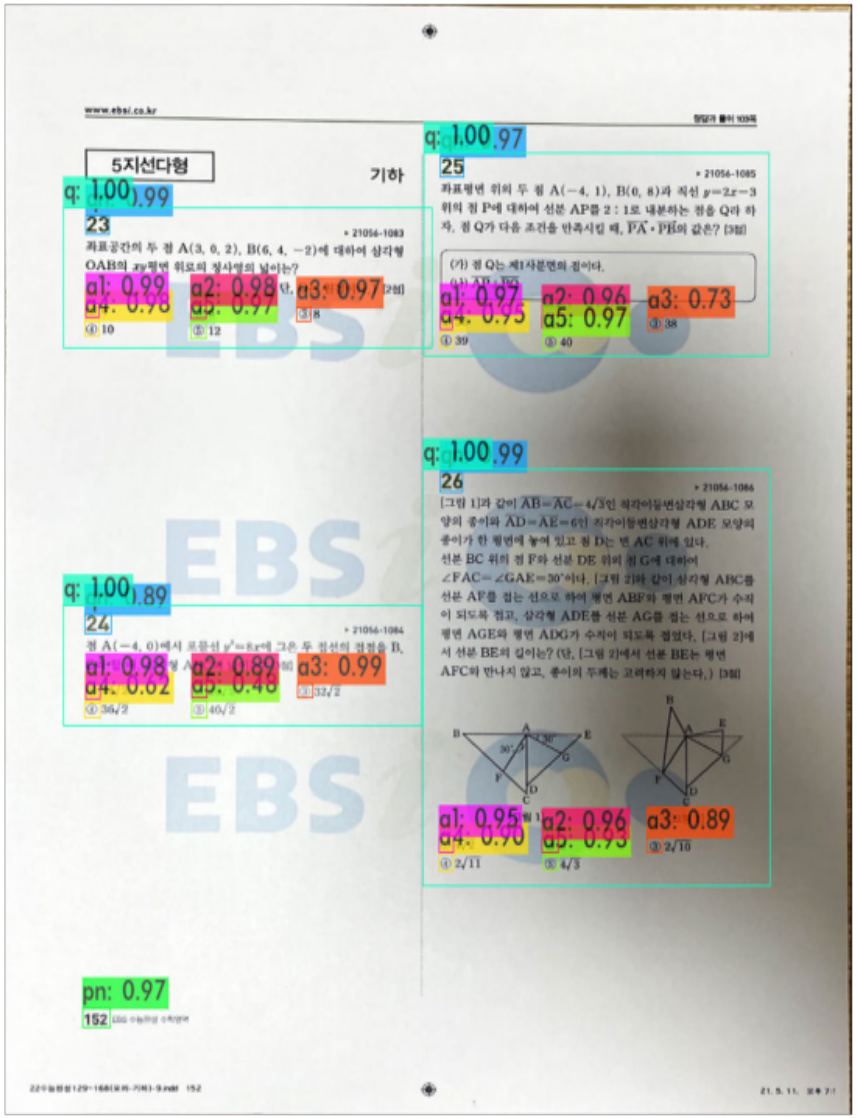
1. `[net]` section
   - `batch=1` - number of samples (images, letters, ...) which will be precossed in one batch
   - `subdivisions=1` - number of mini_batches in one batch, size `mini_batch = batch/subdivisions`, so GPU processes `mini_batch` samples at once, and the weights will be updated for `batch` samples (1 iteration processes `batch` images)
   - `width=416` - network size (width), so every image will be resized to the network size during Training and Detection
   - `height=416` - network size (height), so every image will be resized to the network size during Training and Detection
   - `channels=3` - network size (channels), so every image will be converted to this number of channels during Training and Detection
   - `inputs=256` - network size (inputs) is used for non-image data: letters, prices, any custom data
   - `max_chart_loss=20` - max value of Loss in the image `chart.png`

```
[net]
# Testing
#batch=1
#subdivisions=1
# Training
batch=64
subdivisions=16
width=416
height=416
channels=3
momentum=0.949
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 16000
policy=steps
steps=12800,14400
scales=.1,.1

#cutmix=1
mosaic=1
```

AI – Evaluation Metrics, Training

```
 (next mAP calculation at 4200 iterations)
 Last accuracy mAP@0.50 = 96.95 %, best = 99.03 %
 4200: 2.148810, 2.258631 avg loss, 0.001000 rate, 5.680183 seconds, 268800 images, 18.105155 hours left
Resizing to initial size: 416 x 416  try to allocate additional workspace_size = 52.43 MB
 CUDA allocate done!

 calculation mAP (mean average precision)...
 Detection layer: 139 - type = 28
 Detection layer: 150 - type = 28
 Detection layer: 161 - type = 28
48
 detections_count = 1243, unique_truth_count = 981
class_id = 0, name = a1, ap = 99.93%      (TP = 121, FP = 3)
class_id = 1, name = a2, ap = 99.04%      (TP = 121, FP = 6)
class_id = 2, name = a3, ap = 98.05%      (TP = 119, FP = 2)
class_id = 3, name = a4, ap = 96.55%      (TP = 118, FP = 8)
class_id = 4, name = a5, ap = 97.92%      (TP = 119, FP = 6)
class_id = 5, name = pn, ap = 99.96%      (TP = 44, FP = 0)
class_id = 6, name = q, ap = 99.99%       (TP = 161, FP = 1)
class_id = 7, name = qn, ap = 100.00%             (TP = 162, FP = 0)

 for conf_thresh = 0.25, precision = 0.97, recall = 0.98, F1-score = 0.98
 for conf_thresh = 0.25, TP = 965, FP = 26, FN = 16, average IoU = 75.12 %
```



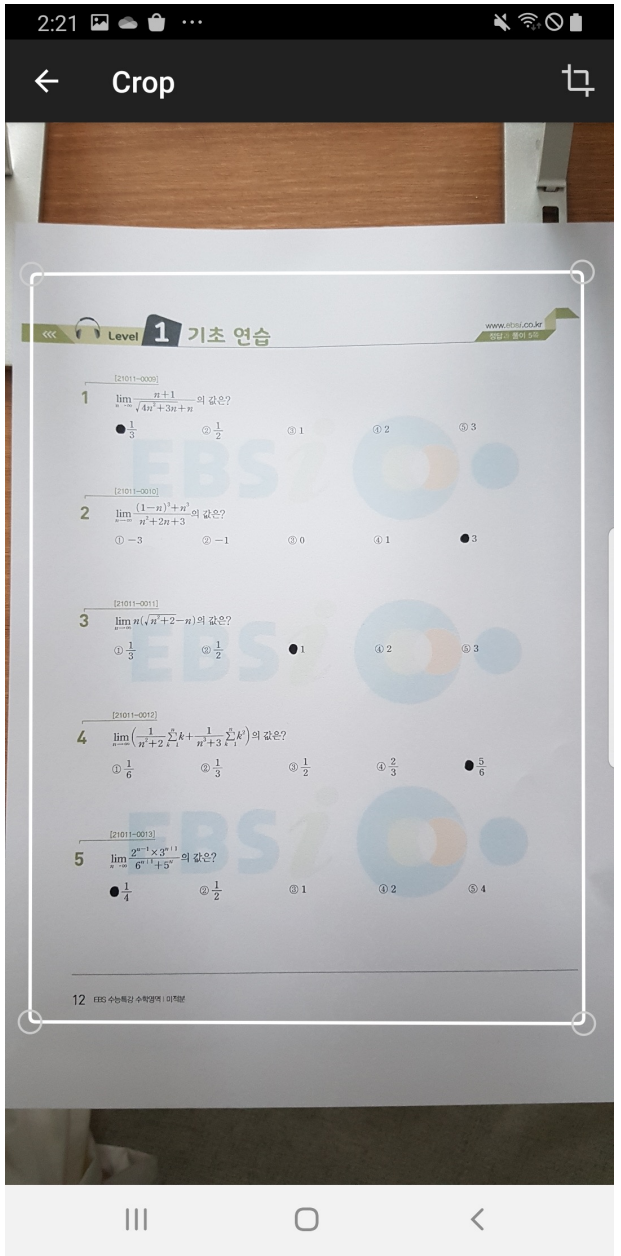

5지선다형
기하
q: 1.00 0.99
23
a1: 0.99 a2: 0.98 a3: 0.97
a4: 0.98 a5: 0.77
q: 1.00 0.97
25
a1: 0.97 a2: 0.96 a3: 0.73
a4: 0.75 a5: 0.97
q: 1.00 0.99
26
a1: 0.95 a2: 0.96 a3: 0.89
a4: 0.90 a5: 0.93
q: 1.00 0.89
24
a1: 0.98 a2: 0.89 a3: 0.99
a4: 0.82 a5: 0.48
pn: 0.97
152


q: 0.93
qn: 0.98
6
a1: 0.97 a2: 0.92 a4: 0.96 a5: 0.93
q: 0.99
qn: 0.97
7
a1: 0.94 a2: 0.97 a3: 0.99 a5: 0.96
q: 0.99
qn: 0.98
8
a1: 0.99 a2: 0.94 a3: 0.98 a4: 0.98
q: 0.99
qn: 0.98
9
a1: 0.99 a3: 0.99 a4: 0.96 a5: 0.97
q: 1.00
qn: 0.98
10
a1: 0.96 a2: 0.77 a3: 0.98
pn: 0.96
47

Implementation – Frontend

Edge_detection library

After taking a picture, user can set the 4 edges of t he box manually to crop the image. Using this technique, user can freely modify the image.



Implementation – Backend

- `POST` /user
- Parameters

Name	Type	Description
`header` user_id	`string`	유저의 로그인 아이디입니다.
`header` user_pw	`string`	유저의 로그인 비밀번호입니다.

- Responses

StatusCode	Values	Description
`200`	`{user_ssd: 12}`	유저의 ssd 정보를 반환합니다.
`409`		이미 존재하는 id입니다.
`500`		

- `GET` /user
- Parameters

Name	Type	Description
`header` user_id	`string`	유저의 로그인 아이디입니다.
`header` user_pw	`string`	유저의 로그인 비밀번호입니다.

- Responses

StatusCode	Values	Description
`200`	`{user_ssd: 12}`	유저의 ssd 정보를 반환합니다.
`401`		등록되지 않은 유저입니다.
`500`		

Implementation - Backend

VULTR

Server	OS	Location	Charges	Status
fancy 1024 MB High Frequency - 141.164.39.229		Seoul	$0.68	Running

```
deinm — root@fancy: /home — ssh root@141.164.39.229 — 80×7
...t@fancy: ~ — ssh root@141.164.39.229   .../home — ssh root@141.164.39.229
[root@fancy:/# cd home
[root@fancy:/home# ls
Capstone_Design_Project  daeyeol  minji
[root@fancy:/home# tmux ls
live: 1 windows (created Wed Nov  3 11:45:02 2021) [142x36]
minji: 1 windows (created Mon Nov  1 13:58:28 2021) [142x36]
root@fancy:/home#
```

Our API server is always running on server with tmux!

Implementation - Backend

32 GB NVMe	64 GB NVMe	80 GB NVMe	128 GB NVMe
$6/month	$12/month	$18/month	$24/month
$0.009/hour	$0.018/hour	$0.027/hour	$0.036/hour
1 CPU	1 CPU	2 CPU	2 CPU
1 GB Memory	2 GB Memory	2 GB Memory	4 GB Memory
1 TB Bandwidth	2 TB Bandwidth	3 TB Bandwidth	3 TB Bandwidth

256 GB NVMe	384 GB NVMe	448 GB NVMe	512 GB NVMe
$48/month	$96/month	$144/month	$192/month
$0.071/hour	$0.143/hour	$0.214/hour	$0.286/hour
3 CPU	4 CPU	6 CPU	8 CPU
8 GB Memory	16 GB Memory	24 GB Memory	32 GB Memory
4 TB Bandwidth	5 TB Bandwidth	6 TB Bandwidth	7 TB Bandwidth

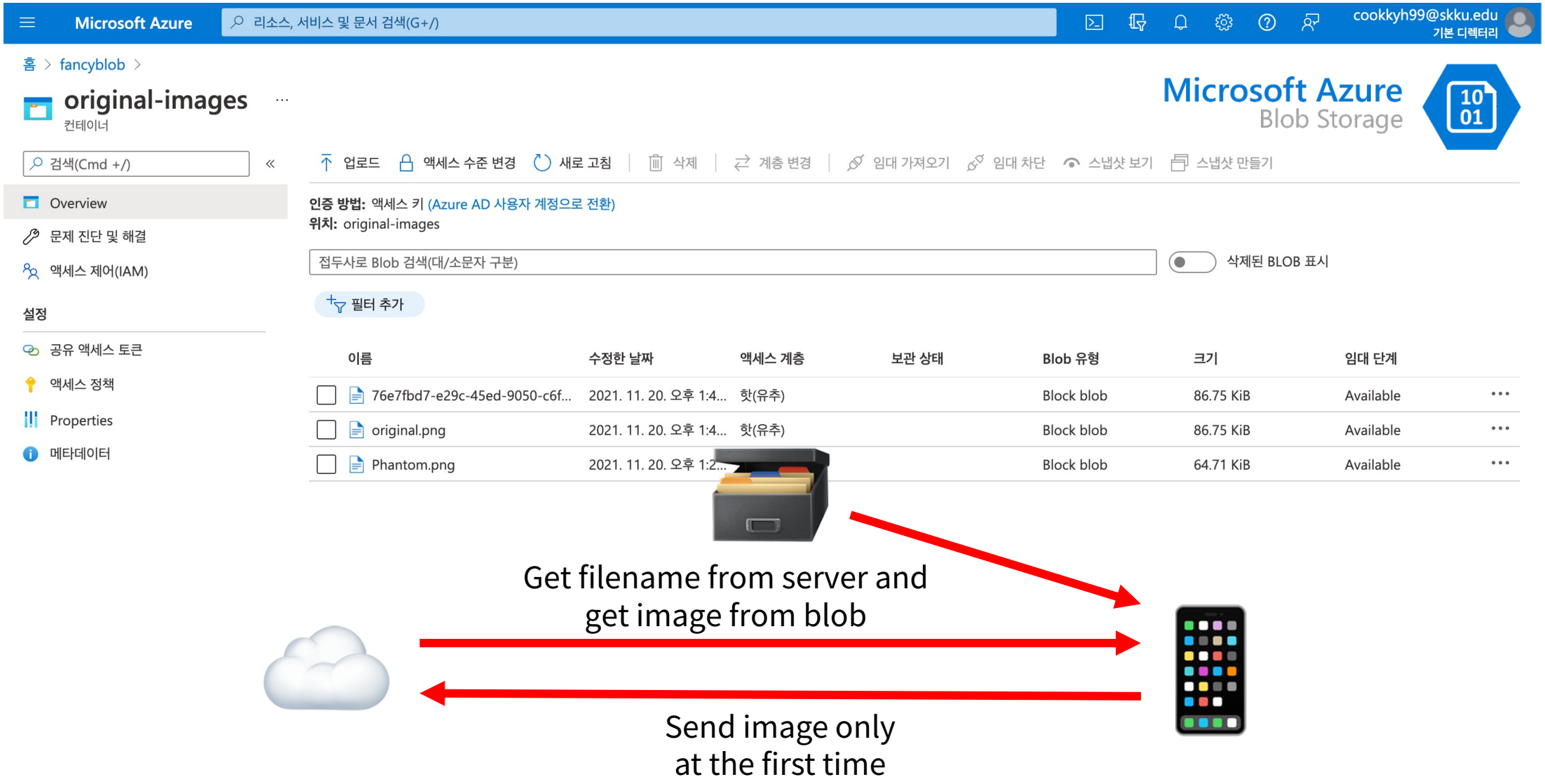
Implementation - Backend

How did we save images?

```
                                    Table "public.problem"
    Column     |            Type             | Collation | Nullable |                  Default
---------------+-----------------------------+-----------+----------+-------------------------------------
 p_id          | integer                     |           | not null | nextval('problem_p_id_seq'::regclass)
 p_description | character varying(200)      |           | not null |
 p_image       | bytea                       |           | not null |
 rn_id         | integer                     |           |          |
 wb_id         | integer                     |           | not null |
 p_created_on  | timestamp without time zone |           |          |
 is_correct    | boolean                     |           | not null |
 pn            | integer                     |           | not null |
 qn            | integer                     |           | not null |
```

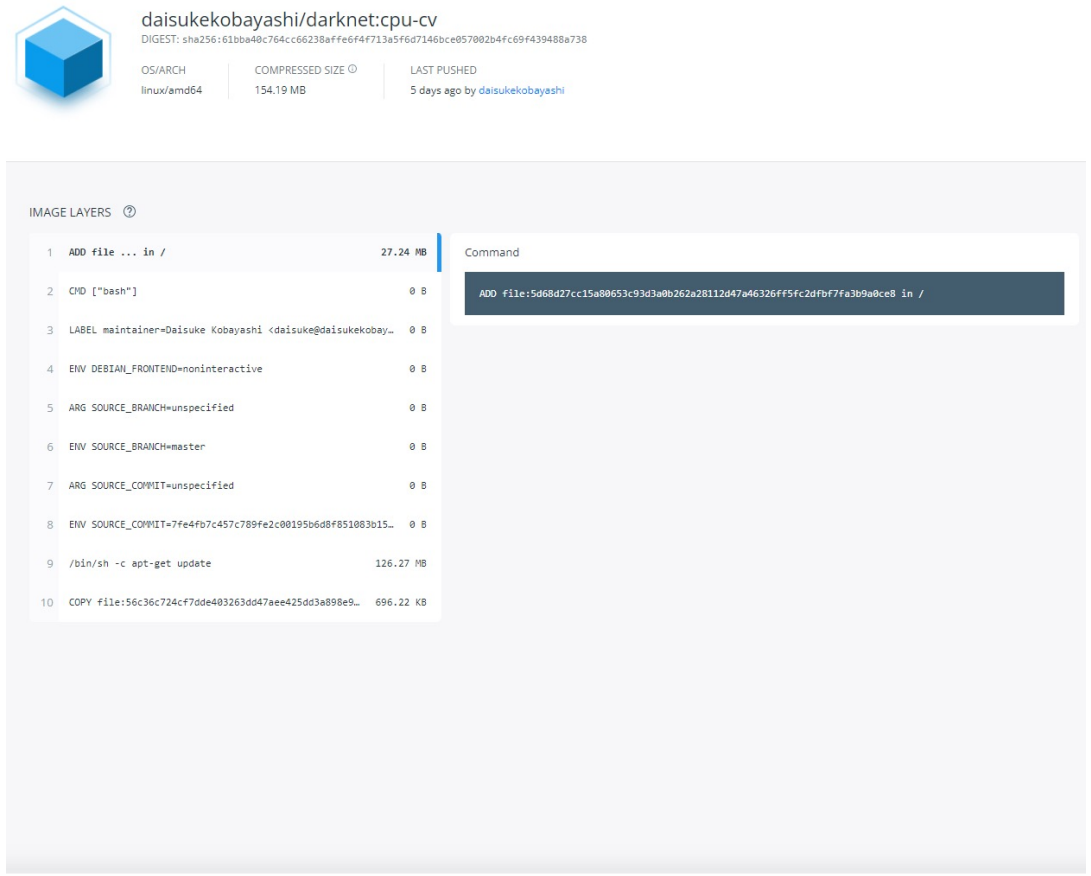
+ should be
decoded/encoded
somewhere every time..

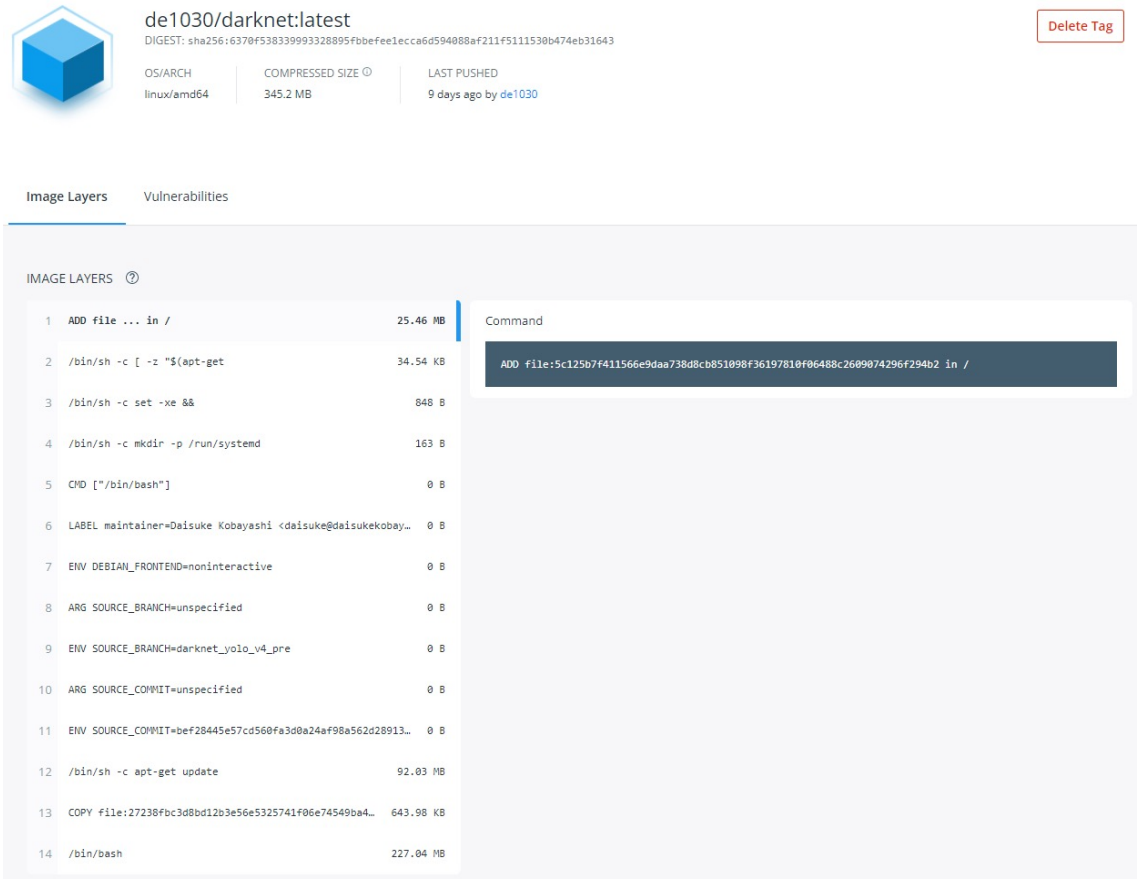
Implementation - Backend



Implementation - AI

How did we run our AI model in server?

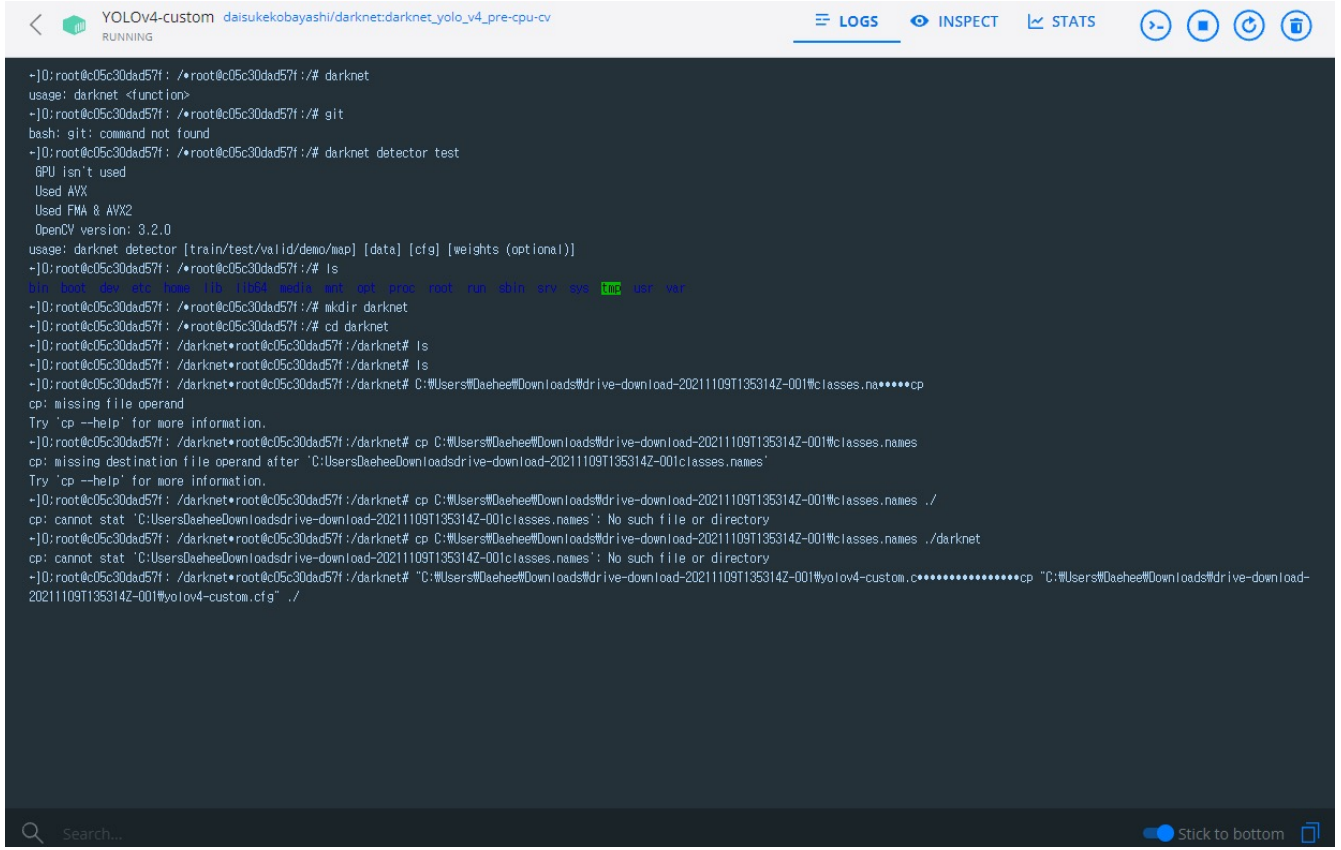




Created a docker container containing the model
Based on darknet image pre-build for cpu use

Implementation - AI

How did we run our AI model in server?



Contains model, learned weigths,
other essential libraries(opencv, tesseract, …)

Implementation - AI

How did we run our AI model in server?

```
def detect_yolo_object(filename, acc):
    os.system('docker start yolo')
    filepath = f"https://fancyblob.blob.core.windows.net/original-images/{filename}"
    os.system(f"wget -P /home/Capstone_Design_Project/AI/YOLO_trained {filepath} -O test.jpg")

    originalimage = cv2.imread("test.jpg")
    editedimage = cv2.resize(originalimage, dsize=(832, 832), interpolation=cv2.INTER_CUBIC)
    cv2.imwrite("test.jpg", editedimage)

    os.system('docker cp test.jpg yolo:darknet/test/test.jpg')
    output_filename = f"./test.sh test/test.jpg {acc}"
    output_json_path = os.path.join(os.getcwd(), "flaskapp", "result.json")
    output_image_path = os.path.join(os.getcwd(),  "flaskapp", "predictions.jpg")
    os.system(f'docker exec yolo sh -c "cd darknet; {output_filename}";docker cp yolo:darknet/result.json {output_json_path};docker cp yolo:darknet/predictions.jpg {output_image_path}')
```

using the system command,
run code in docker, retrieve the result
and post-process the inference result

Implementation - AI

Auto grading

1) Recognize “qn” and “pn” with tesseract

Based on the detection result with yolo,
get ocr result by tesseract

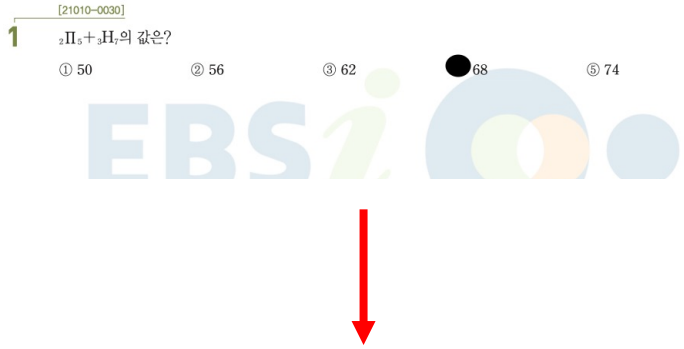
2) Detect user’s answer

“not detected” answer option is the answer
that user has selected

3) Get answer from database

Implementation - AI

Problem Regeneration



${ }_{2} \Pi_{5}+{ }_{3} \mathrm{H}_{7}$ 의 값은?

$$ {}_4\Pi_9 + {}_1\mathrm{H}_4 \text{ 의 값은?} \quad {}_7\Pi_8 + {}_1\mathrm{H}_3 \text{ 의 값은?} $$

$$ {}_7\Pi_8 + {}_6\mathrm{H}_8 \text{ 의 값은?} $$

Demo url :
http://158.247.200.51:443/regenerate_problem/98

1) Crop detected question area

2) Convert question into latex

Using mathpix, convert into latex.
Save the result in database.

3) Regenerate question

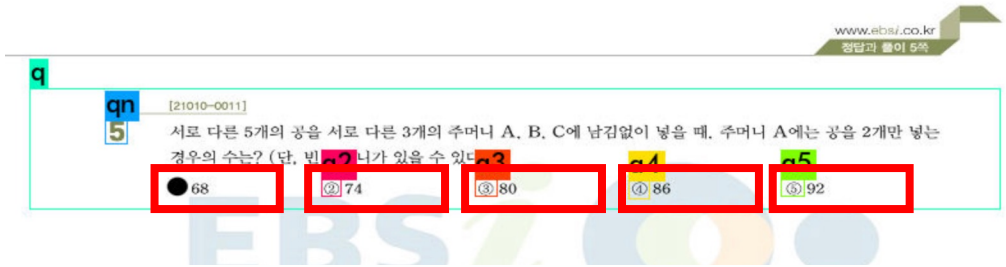
Detect numbers in $$ and randomize variables.

4) Render result as html

To render latex, we used katex library in javascript.

Implementation - AI

Shuffle answer options



1) Crop answer options

Based on YOLO result, crop answer options.
Had implemented some algorithms to crop options
Even some of the options are not detected by YOLO.
Save image in blob.

2) Shuffle options and grading

From frontend side, shuffle the options and
Show the options to the user.
Grade the answer by getting the correct answer from API.

Limitations/Challenges

Answers should be saved in the database

Through discussion, we concluded that extracting answers from the answer sheet is difficult. So we decided to save answers in database in advance.

It is practically not possible to enter all the answers of the existing workbooks in advance. But by this implementation, we could make users only take pictures of the problem page. Also, since there are few main famous workbooks that student use for studying(쎈, 개념원리, 수능특강, …), maybe this might be even practical for some point of view. We don’t have to run model again and again to grade the same page because we save the answer data in database.

Limitations/Challenges

Accuracy of model

With good image with good conditions, our model worked well.
But if there was a 3d-rolling on the image, the performace of the model and the result of OCR was so bad.

Also, if user writes something on the workbook, the question area was detected according to the user's handwriting rather than the printed text. So the question-question number matching was not working well.

We think that we can solve those problems by training model with additional datasets - 1) images with 3d rolling augmentation and 2) images which has handwritten letters on it.

Evaluations

Good UI/UX according to simple testing 💯

Due to the tight development time, it was quite difficult to recruit many test users. However, when we asked for some people to test our application, they answered the overall UI/UX of our app was good since it was simple and not complicated.

Achieved our initial objective 🎉

Out objective was to making a better review note applications compared to existing ones. We implemented auto grading function, problem regenerating function, and randomizing answer options function to achieve this goal. In this point of view, our product meets the initial objective of the project.

Thanks