

Image-based Pill Identification Service

Using Artificial Intelligence



팀원 소개



강주애

역할: 서버 & 웹 아키텍처 구축
보조: 데이터 베이스 구축



박민영

역할: 시스템 아키텍처 & AOS 구현
보조: 서버 & 웹 아키텍처 구축



정효인

역할: 이미지 기반 ML 알고리즘 모델링
보조: 데이터베이스 구축

INDEX

- 01 서비스 기획 및 개요
Service Planning and Overview
- 02 개발 환경
Development Environment
- 03 구현 방식
How we implement the service
- 04 시연
Demonstration
- 05 결론 및 개선점
Conclusion and Improvement plan

서비스 기획 배경 01

이미지 기반 알약 식별 서비스가 필요한 이유

왜 필요해?

Q. 약국이 모두 닫은 저녁 9시, 하필 만성 위염 환자인 A씨에겐 남은 위염약이 없고, A씨의 약품보관함엔 길을 잃은 약들 뿐이다.
다음 중 A씨가 섭취해야 할 약은?



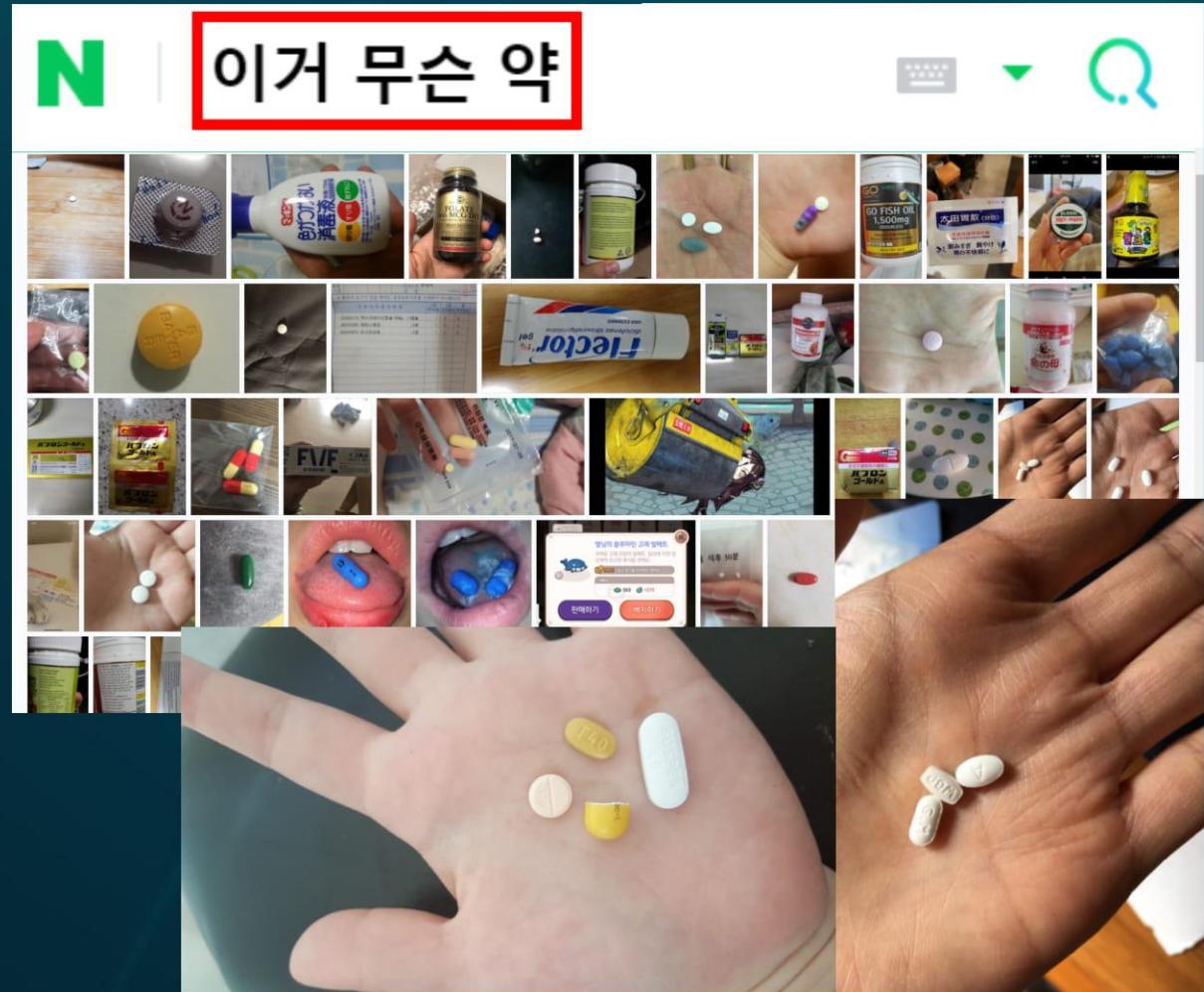
Q. 약국이 모두 닫은 저녁 9시, 하필 만성 위염 환자인 A씨에겐 남은 위염약이 없고, A씨의 약품보관함엔 길을 잃은 약들 뿐이다.
다음 중 A씨가 섭취해야 할 약은?



1. 구내염
2. 변비약
3. 다래끼
4. 위염

의약품 정보접근성 실태 |

정확한 정보를 요하는 의약품의 경우,
인가되지 않은 일반 포털 검색 결과는
위험을 초래할 수 있음



의약품 정보접근성 실태 II

약물 오용시, 약화(藥禍) 사고와 같은
부작용 발생 가능성 증가

한겨레 | 사회
의료·건강

인터넷한겨레 | 씨네21 | 한겨레21 | 이코노미
<http://www.hani.co.kr> | **한토마** | [사설·칼럼](#) | [정치](#) | [경제](#) | [사회](#) | [스포츠](#) | [국제](#) | [문화](#) | [과학](#)
[사건·판결](#) | [노동](#) | [교육](#) | [여성](#) | [행복의창](#) | [건강](#) | [환경](#) | [지역](#) | [장애인](#) | [웃긴소식](#) | [인사](#)

집안내 방치약 '어림짐작 복용' 많아

약사회 조사...감기약·영양제등 바로알고 먹어야

집안에 굴러다니는 약을 어림짐작해서 먹거나 바르는 사람이 절반 가까이나 되는 것으로 나타났다. 이 경우 약화(藥禍) 사고가 빚어질 가능성이 있어 각별한 주의가 필요하다.

각 가정에 쓰다남은 약은 평균 3만9천 원 정도 어치인 것으로 집계됐다. 이중 66% 정도는 어디에 사용하는 약인지 정확한 구별조차 되지 않는다. 버리기 아까워 방치하다 대충 복용하는 오남용의 개연성이 높은 대목이다.

대한약사회가 창립 50주년을 맞아 서울 거주 시민 192명을 대상으로 최근 설문조사한 데 따르면 응답자의 47.8%가 '보관중인 의약품을 정확한 정보없이 대강의 짐작으로 복용한 적이 있다'고 밝혔다.

▲ 기사 출처 : <http://legacy.www.hani.co.kr/section-005100031/2004/10/005100031200410310938001.html>

대한 약사회에서 서울 거주 시민
192명에게 물었다.

63.4%

Q. 가정에서 보관 중인 약의 용법을 아시나요?
A. 잘 모르겠습니다.

47.8%

Q. 보관중인 의약품을 정확한 정보 없이
대강의 짐작으로 복용한 적이 있나요?
A. 네 있습니다.

불평등을 발견하다



일년간 폐기되는 약의 규모

건강보험심사평가원에 따르면, 한 해 폐기되는 의약품의 규모는 394,000kg(약 2180억 원) 수준

홈 > 정책 > 심평원공단

2016년 한해만 급여약 2180억원 어치 버려졌다

▲ 최은택 | Ⓛ 입력 2019.01.11 06:06 | Ⓜ 댓글 0



| 심평원, 자체 연구...중복처방 미사용 비용 1382억원 추정

[낭비되는 의약품 규모 등 분석 연구]

한해 버려지는 급여의약품 비용이 같은 해 전체 외래 원외처방 비용의 1.8%를 점유한다는 연구결과가 나왔다. 2016년에 적용해 환산하면 2000억원이 넘는 액수다. 또 같은 성분의 약품이 중복 처방돼 역시 사용되지 못하는 의약품도 같은 해 기준 1400억원에 달하는 것으로 추정됐다.

이 같은 사실은 심사평가원이 자체 연구한 '낭비되는 의약품 규모, 비용 및 요인 분석 연구: 미사용으로 버려지는 처방전약 중심으로' 보고서(연구책임자 김지애 부연구위원)를 통해 확인됐다. 공동연구자로는 이혜영 주임연구원, 문경준 주임연구원, 박혜경 성균관대 연구교수 등이 참여했다. 일반국민이 보고하는 국내 의약품 낭비규모와 비용에 관한 구체적인 정보를 제공하는 최초의 연구다.

연구진은 낭비되는 의약품 규모를 파악하기 위해 '중복처방으로 인한 미사용 규모 및 비용', '낭비되는 의약품 규모 및 비용'을 각각 산출했다.

▲ 기사 출처 : <http://www.hitnews.co.kr/news/articleView.html?idxno=5652>

의약품 정보접근성 실태 III

현재 '약학정보원'만이 유일한 의약품 검색 경로이며, 그마저도 최대 8개의 정보 입력을 유도

식별 정보 입력

식별도우미
알약에 있는 글자나 숫자, 그림(마크)을 입력하세요.
예시) A B KPI0 ▶ 이 알약의 경우
• 식별문자에 AB 입력
• 식별마크에서 KPI0 선택

식별문자	1 문자1 <input type="text"/> 일치 문자2 <input type="text"/> 일치
식별마크	2 마크 선택하기
제형	3 정제 <input checked="" type="radio"/> 경질캡슐 <input type="radio"/> 연질캡슐 <input type="radio"/> 기타 <input type="radio"/> 전체
모양	4 원형 <input checked="" type="radio"/> 타원형 <input type="radio"/> 장방형 <input type="radio"/> 반원형 <input type="radio"/> 삼각형 <input type="radio"/> 사각형 <input type="radio"/> 마름모형 <input type="radio"/> 오각형 <input type="radio"/> 육각형 <input type="radio"/> 팔각형 <input type="radio"/> 기타 <input type="radio"/> 전체
색상	5 하양 <input checked="" type="radio"/> 노랑 <input type="radio"/> 주황 <input type="radio"/> 분홍 <input type="radio"/> 빨강 <input type="radio"/> 갈색 <input type="radio"/> 연두 <input type="radio"/> 초록 <input type="radio"/> 청록 <input type="radio"/> 파랑 <input type="radio"/> 남색 자주 <input type="radio"/> 보라 <input type="radio"/> 회색 <input type="radio"/> 검정 <input type="radio"/> 투명 <input type="radio"/> 기타 <input type="radio"/> 전체
분할선	6 없음 <input checked="" type="radio"/> (+)형 <input type="radio"/> (-)형 <input type="radio"/> 기타 <input type="radio"/> 전체

▲ 현행 서비스(약학정보원)

식별표시 신고센터

다시 입력 C 검색

의약품 정보, 누구에게나 평등하게

‘이미지 기반
의약품 인식 서비스’

개발 환경

02

서비스 구현을 위한 개발 일정 및 환경 구성

Front-End

HTML



JS



CSS



Back-End

[웹&어플리케이션]



Android
Studio



[공통 개발환경]



[AI & ML]



OpenCV



Tesseract OCR

[서버 & DB]

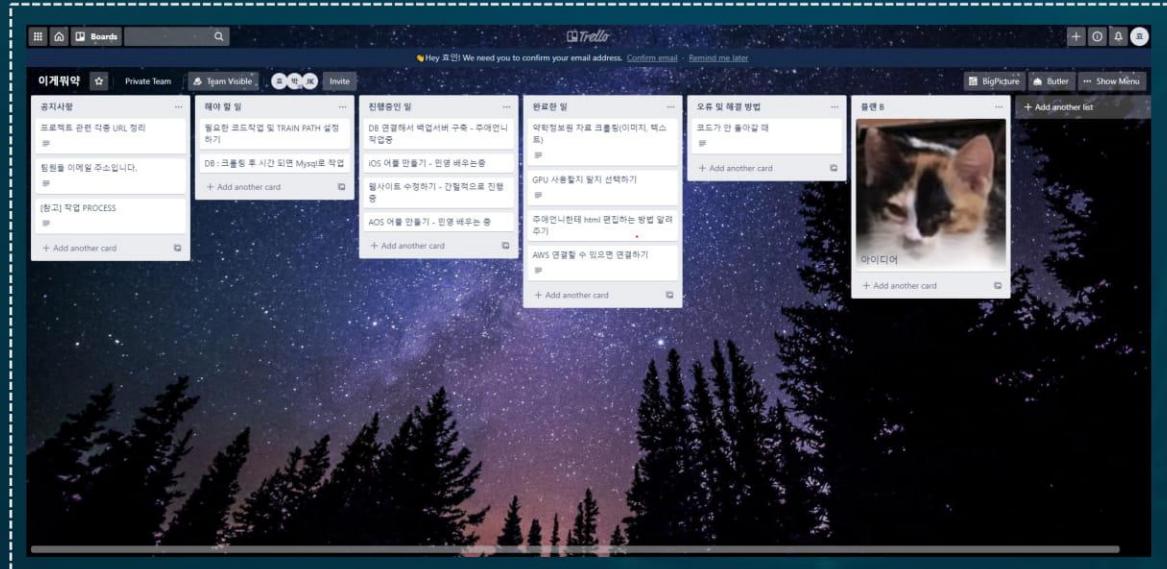


개발 환경 – 업무 진행

‘ Work Breakdown Structure ’

Project	미개발		상주인		
Manager	정효인		광주예		
Writer	정효인		박민영		
Version	1.1				
Issued date	2020-10-23				
Work Break-down Structure					
Procedures	Steps	Tasks	담당자	schedule	산출물/비고
			시작일	종료일	
1.0. 분석					
1.1.0. 시장 현况 분석, 브레인 스토밍				9월 12일	9월 19일
1.2.0. 요구사항 분석				9월 12일	9월 19일
2.0. 설계					
2.1.0. DB 설계					
2.1.1. AWS 앤스턴드/RDS 생성(프리티어)			정효인	9월 19일	10월 10일
2.1.2. 미학 정보 설계/설계			박민영	9월 19일	10월 10일
2.2.0. ML 모델 설계					
2.2.1. 이미지기반 ML 논문 탐색			박민영	9월 19일	10월 10일
2.2.2. OpenCV 알고리즘 설계			광주예	10월 17일	10월 24일
2.2.3. OCR 알고리즘 설계			정효인	10월 17일	10월 24일
2.3.0. 템/업 설계					
2.3.1. 메인 디자인 설계 및 컨셉			광주예	10월 17일	미정
2.3.2. UI 및 퀄리티 설계 및 컨셉			광주예	10월 17일	미정
2.4.0. 시스템 아키텍처 설계					
2.4.1. 웹 아키텍처 설계(WAS, DB)			박민영	10월 17일	10월 24일
2.4.2. 인프라 아키텍처 설계(AWS)			박민영	10월 17일	10월 24일
2.4.3. 서비스 품질우수 설계(업무 측면)			광주예	10월 17일	10월 24일
3.0. 개발/구현					
3.1.0. DB 구현					
3.1.1. 미학 정보 DB 구축(aws 틀 활용)			정효인	9월 19일	9월 26일
3.1.2. 미학 정보 DB 생성(AWS로)			박민영	10월 17일	10월 24일
3.2.0. ML 모델 구현					
3.2.1. 모델 퓨사리즘 진행 및 히不合适 모델 신청			정효인	10월 17일	미정
3.2.2. Custum 모델 생성 및 학습			박민영	10월 17일	미정
3.3.0. 템/업 구현					
3.3.1. 메인 시연 작업			광주예		
3.3.2. 전체 퀄리티 디자인			광주예		
4.0. 테스트					
4.1.0. 단일 테스트					
4.1.1. ML Custom 모델 성능 테스트					
4.1.2. AWS/DynamoDB 테스트					
4.1.3. 템/업 퀄리티 테스트					
4.2.0. 통합 테스트					
4.2.1. 테스트 케이스 진행(시나리오 기반)					
5.0. 통합					
5.1.0. 통합 서비스					
5.1.1. 모로보타일 서비스 진행					
5.1.2. 최종 검수 및 수정사항 진행					
5.2.0. 발표 준비					
5.2.1. 발표 자료 준비					
6.0. 발표					
6.1.0. 발표					

‘Trello’

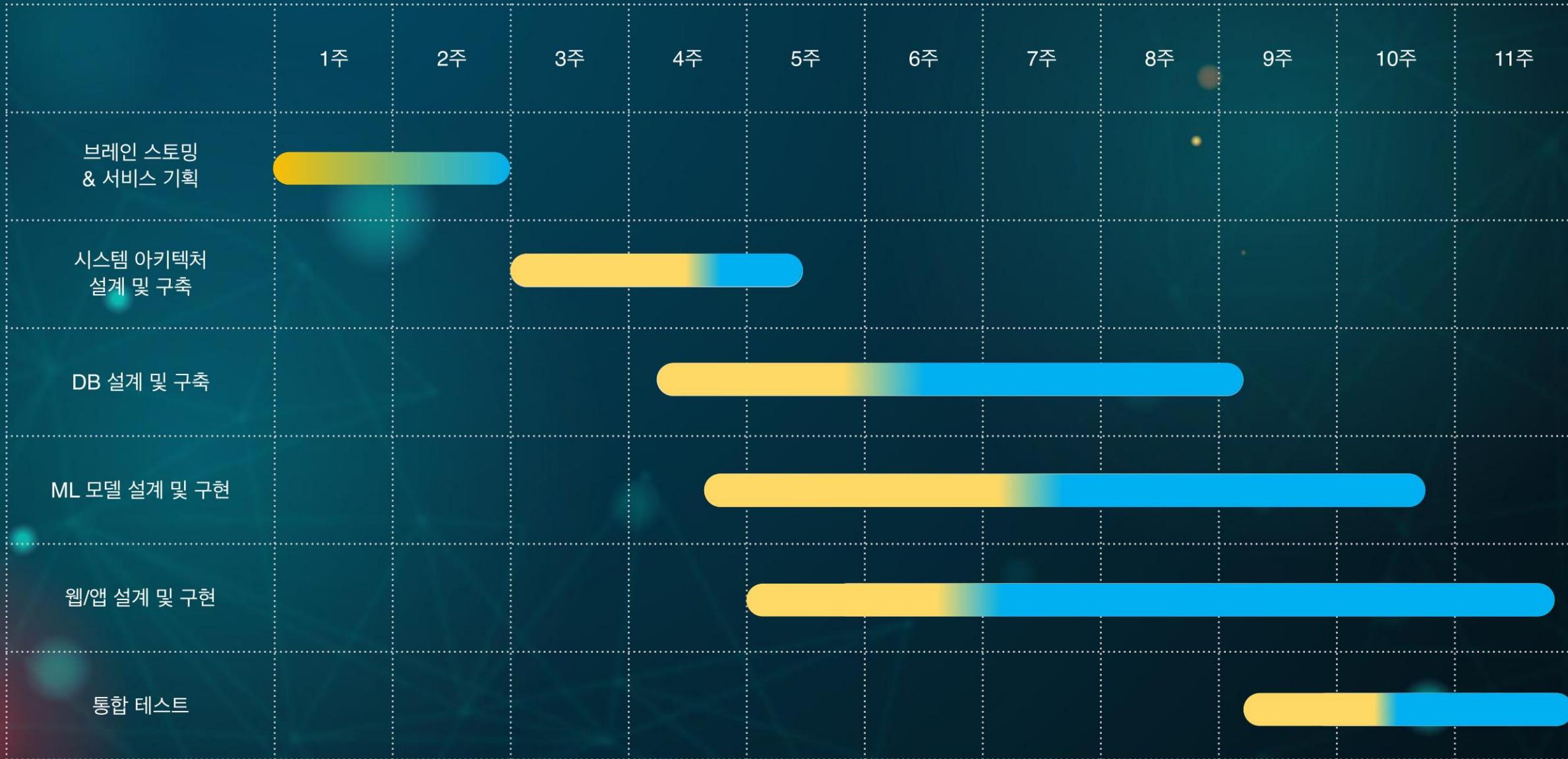


▲ 유연한 업무 공유를 위한 도구 사용

◀ 일정 관리를 위한 프로젝트 일지 작성 및 문서화(~Ver.1.7)

개발 환경 – 일정

설계 단계
구현 단계



구현 방식

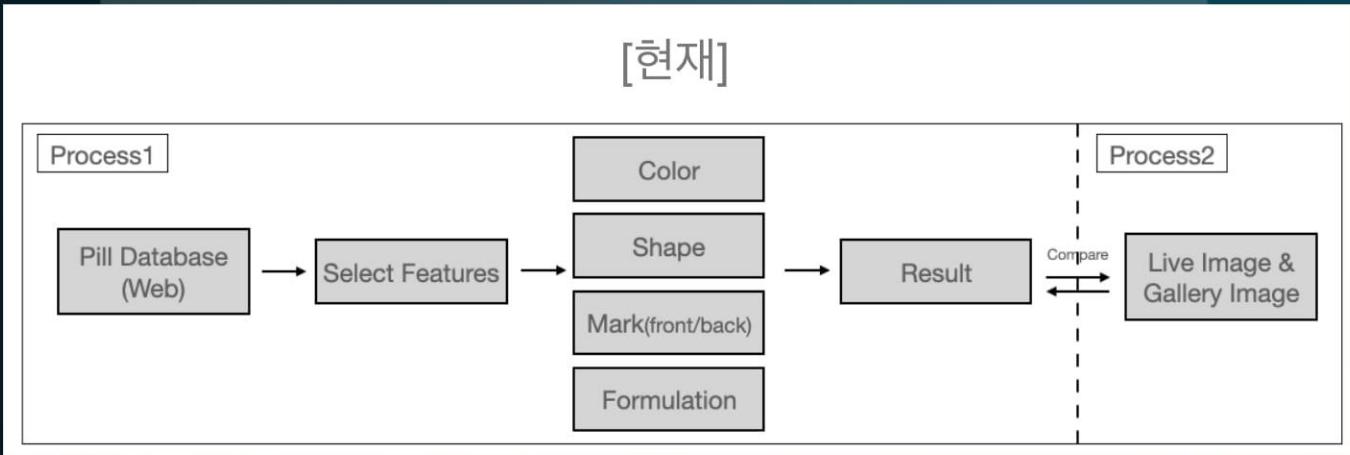
03

새로운 서비스 흐름을 제안하고
세 단계의 설계 방식으로 구성

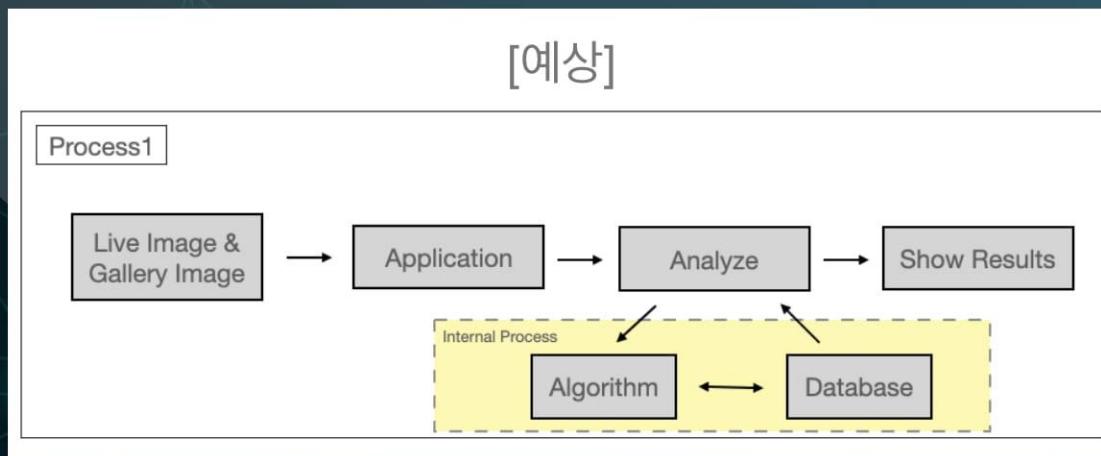
새로운 서비스 흐름 제안

기존의 복잡한 검색 시스템에서
벗어나 이미지 기반으로 편의성
증대

[현재]



[예상]



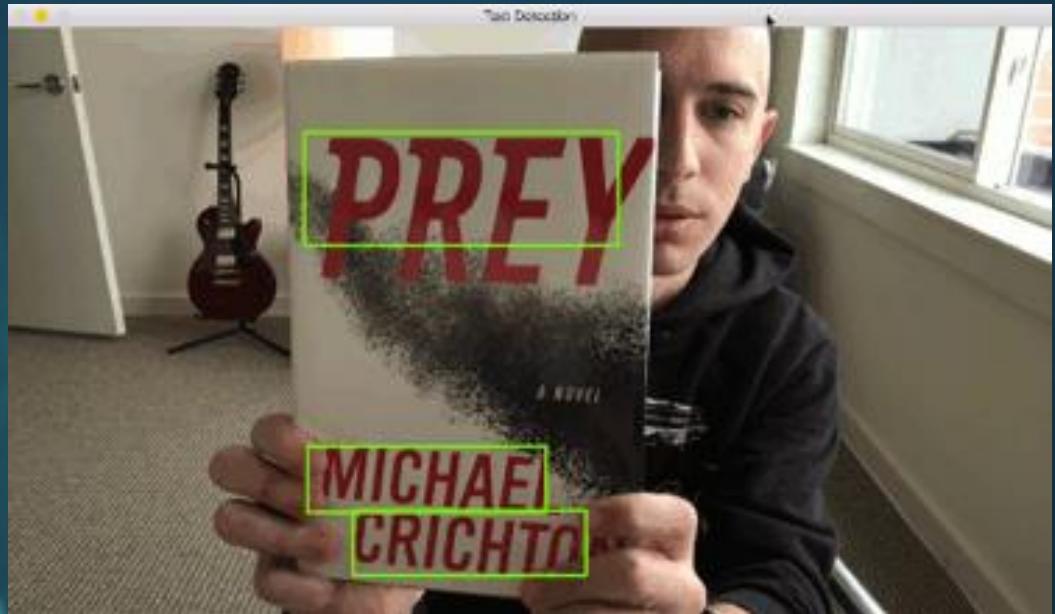
알약 식별 서비스 아키텍처 구조도



ML 알고리즘 구현



알약 식별 알고리즘 모델링



문자영역 인식을 위한
OpenCV – EAST 모델

`used_model = frozen_east_text_detection.pb`

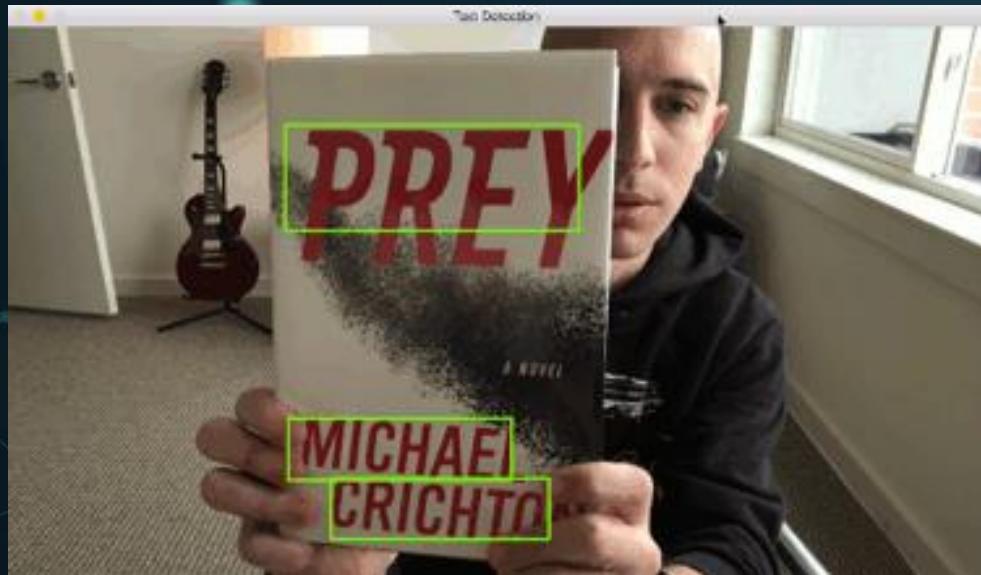
`height = 480`

`width = 480`

`padding = 0.06`

`min_confidence=0.5`

알약 식별 알고리즘 모델링



“ 이미지의 문자 영역을 인식하는
OpenCV - EAST 모델 ”

“ 학습된 EAST 모델을
알약에 적용시킨 모습 ”

알약 식별 알고리즘 모델링



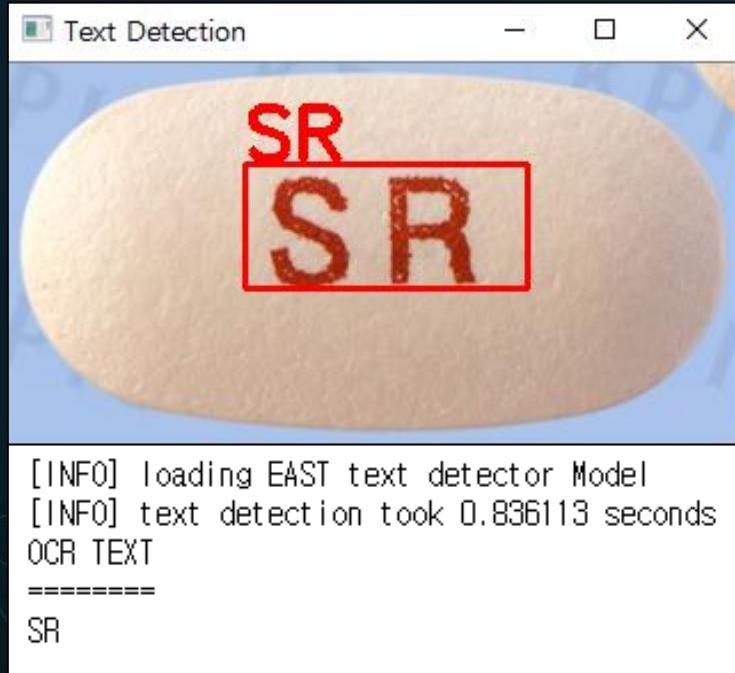
#Tesseract-OCR의 예시

이미지 내의 글자 추출을 위한
Tesseract - OCR

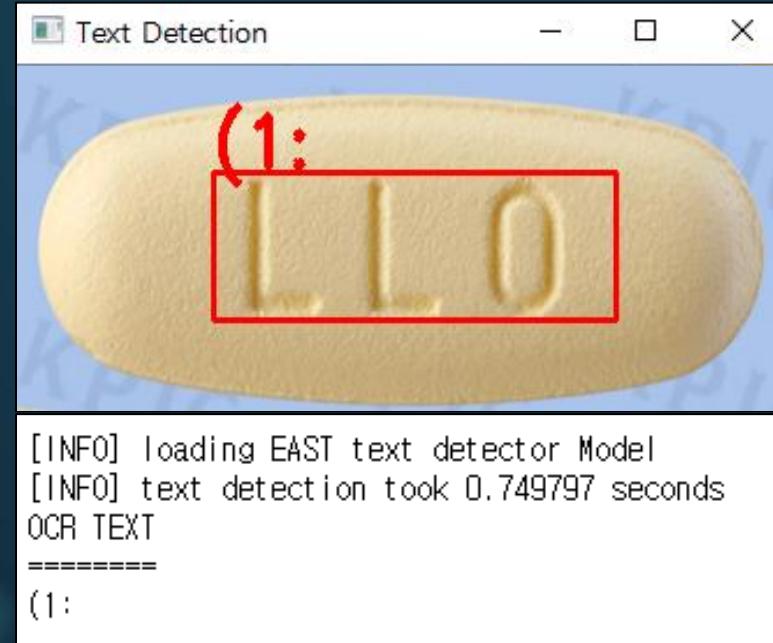
used_version = pytesseract 4.0

config = ("-l eng+kor --oem 1 --psm 7")

알약 식별 알고리즘 모델링

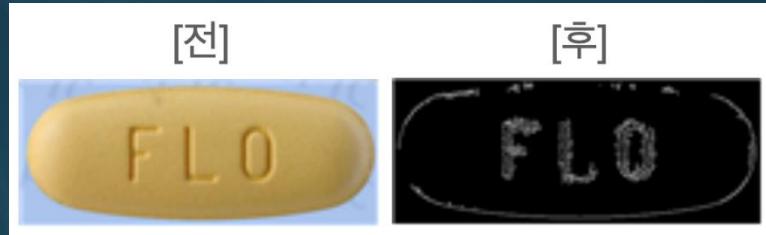


“ EAST모델을 통해 인식된 문자 영역을
OCR로 읽어내는 모습 ”



“ 하지만 식별 코드가 음각으로 된
알약은 추가적인 전처리가 필요 ”

알약 식별 알고리즘 모델링



| 캐니-엣지 기법



| 가우시안 필터



식별 코드가 음각으로 된 알약에
추가적인 이미지 전처리 진행

음각 이미지 100개를 테스트하여,
각각의 성능을 측정



| 모폴로지 기법

”

알약 식별 알고리즘 모델링



| 캐니-엣지 기법

3%



| 가우시안 필터

43%



| 모폴로지 기법

12%

알약 식별 알고리즘 모델링

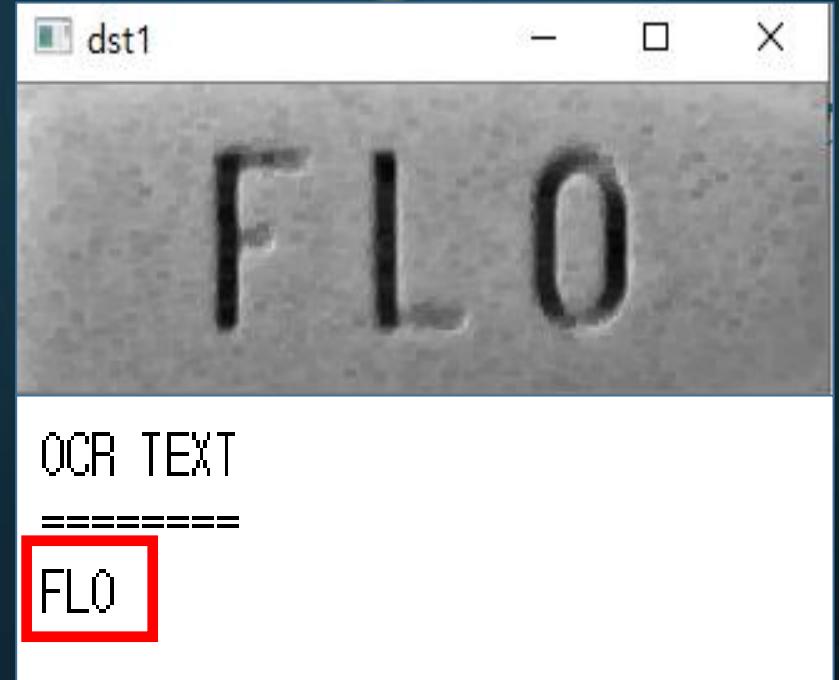
| 가우시안 필터



| 모폴로지 기법



“가우시안 + 모폴로지를 활용하여
이미지 전처리 진행”



“전처리 후 성공적으로
문자를 읽어내는 모습”

알약 식별 알고리즘 구현 코드

```
import numpy as np
import time
import cv2
from utils.object_detection import non_max_suppression
import pytesseract

img = "./img/119.jpg"
east = "C:/Coding/python/openCV/EAST/frozen_east_text_detection.pb"
height = 480 # in model .pb, model works in any pb with width and height multiple of 32
width = 480
padding = 0
minConfidence = 0.5 # probability threshold

# loading image and getting image dimension
image = cv2.imread(img)
imageH, imageW = image.shape[:2]
cv2.imshow("Input Image", image)
cv2.waitKey(0)

# set the new width and height and then determine the ratio to change for both
# will be used to revert size to the original size
rH = imageH / float(height)
rW = imageW / float(width)

# resize the image and prep the new dimension
image_resized = cv2.resize(image, (width, height))
cv2.imshow("Resized Image", image_resized)
cv2.waitKey(0)

# load pre-trained EAST text detector NN Model
print("[INFO] loading EAST text detector Model")
net = cv2.dnn.readNet(east)
net.getLayerNames()

layerNames = [net.getLayerNames()[-1], # 'feature_fusion/Conv_7/Sigmoid'
              net.getLayerNames()[-3]] # 'feature_fusion/concat_3'

blob = cv2.dnn.blobFromImage(image_resized,
                             scalefactor = 1.0,
                             size = (width, height),
                             mean=(123.68, 116.78, 103.94), # mean pixel intensity across all training images
                             swapRB = True,
                             crop = False)

start = time.time()
net.setInput(blob)
scores, geometry = net.forward(layerNames)
end = time.time()

# show timing info on text detection
print("[INFO] text detection took {:.6f} seconds".format(end - start))

numRows, numCols = scores.shape[2:4]
rois = []
confidences = []

for y in range(0, numRows):

    scoresData = scores[0, 0, y]
    xData0 = geometry[0, 0, y]
    xData1 = geometry[0, 1, y]
    xData2 = geometry[0, 2, y]
    xData3 = geometry[0, 3, y]
    anglesData = geometry[0, 4, y]

    # loop over number of columns
    for x in range(0, numCols):
        # if score less than confidence threshold ignore it
        if scoresData[x] < minConfidence:
            continue
```

```
        offsetX, offsetY = (x * 4.0, y * 4.0)
        h = xData0[x] + xData2[x] # upper/lower
        w = xData1[x] + xData3[x] # left/right

        angle = anglesData[x]
        sin = np.sin(angle)
        cos = np.cos(angle)

        endX = int(offsetX + (cos * xData1[x]) + (sin * xData2[x]))
        endY = int(offsetY - (sin * xData1[x]) + (cos * xData2[x]))
        startX = int(endX - w)
        startY = int(endY - h)

        # add the bounding box coordinates and probability score to the respective lists
        rois.append((startX, startY, endX, endY))
        confidences.append(scoresData[x])

    boxes = non_max_suppression(np.array(rois), probs=confidences)

results = []

# loop over the bounding boxes
for (startX, startY, endX, endY) in boxes:
    startX = int(startX * rW)
    startY = int(startY * rH)
    endX = int(endX * rW)
    endY = int(endY * rH)

    dx = int((endX - startX) * padding)
    dy = int((endY - startY) * padding)

    # apply padding to each side of the bounding box respectively
    startX = max(0, startX - dx)
    startY = max(0, startY - dy)
    endX = min(imageW, endX + (dx * 2))
    endY = min(imageH, endY + (dy * 2))

    # extract the actual padded roi "Region of Interest"
    roi = image[startY:endY, startX:endX]

    config = ("-l eng+kor --oem 1 --psm 7")
    text = pytesseract.image_to_string(roi, config=config)
    #text = pytesseract.image_to_string(roi, lang='eng+Hangeul')
    # add the bounding box coordinates and OCR'd text to the list of results
    results.append((startX, startY, endX, endY, text))

results = sorted(results, key=lambda r:r[0][1])

# loop over the results
for ((startX, startY, endX, endY), text) in results:
    # display the text OCR'd by Tesseract
    print("OCR TEXT")
    print("=====")
    print("{}\n".format(text))

    # strip out non-ASCII text so we can draw the text on the image using OpenCV,
    # then draw the text and a bounding box surrounding the text region of the input image
    text = ''.join([o if ord(o) < 128 else '' for o in text]).strip()
    output = image.copy()
    cv2.rectangle(output, (startX, startY), (endX, endY), (0, 0, 255), 2)
    cv2.putText(output, text, (startX, startY - 5), cv2.FONT_HERSHEY_SIMPLEX, 1.2, (0, 0, 255), 3)

    # show the output image
    cv2.imshow("Text Detection", output)
    cv2.waitKey(0)

cv2.destroyAllWindows()
```

알약 식별 알고리즘 구현 코드

```
import numpy as np
import time
import cv2
from imutils.object_detection import non_max_suppression
import pytesseract
import os

img = "./img/119.jpg"
east = "C:/coding/python/openCV/EAST/frozen_east_text_detection.pb"
height = 480 # in model do, model works in any pic with width and height multiple of 32
width = 480
padding = 0
minConfidence = 0.5 # probability threshold

# loading image and getting image dimension
image = cv2.imread(img)
imageH, imageW = image.shape[:2]
cv2.imshow("Input Image", image)
cv2.waitKey(0)

# set the new width and height and then determine the ratio i change for both
# will be used to revert size to the original size
rH = imageH / float(height)
rW = imageW / float(width)

# resize the image and prep the new dimension
image_resized = cv2.resize(image, (width, height))
cv2.imshow("Resized Image", image_resized)
cv2.waitKey(0)

# load pre-trained EAST text detector Model
print("[INFO] loading EAST text detector Model")
net = cv2.dnn.readNet(east)
net.getLayerNames()

layerNames = [net.getLayerNames()[-1], # 'feature_fusion/Conv_7/Sigmoid'
              net.getLayerNames()[-3]] # 'feature_fusion/concat_3'

blob = cv2.dnn.blobFromImage(image_resized,
                             scalefactor = 1.0,
                             size = (width, height),
                             mean=[123.68, 116.78, 103.94], # mean pixel intensity across all training images
                             swapRB = True,
                             crop = False)

start = time.time()
net.setInput(blob)
scores, geometry = net.forward(layerNames)
end = time.time()

# show timing info on text detection
print("[INFO] text detection took {:.6f} seconds".format(end - start))

numRows, numCols = scores.shape[2:4]
results = []
confidences = []

for y in range(0, numRows):
    scoresData = scores[0, 0, y]
    geometry0 = geometry[0, 0, y]
    geometry1 = geometry[0, 1, y]
    geometry2 = geometry[0, 2, y]
    geometry3 = geometry[0, 3, y]
    anglesData = geometry[0, 4, y]

    # loop over number of columns
    for x in range(0, numCols):
        # if score less than confidence threshold ignore it
        if scoresData[x] < minConfidence:
            continue
```

```
        offsetX, offsetY = (x * 4.0, y * 4.0)
        h = xData0[x] + xData2[x] # upper/lower
        w = xData1[x] + xData3[x] # left/right

        angle = anglesData[x]
        sin = np.sin(angle)
        cos = np.cos(angle)

        endX = int(offsetX + (cos * xData1[x]) + (sin * xData2[x]))
        endY = int(offsetY - (sin * xData1[x]) + (cos * xData2[x]))
        startX = int(endX - w)
        startY = int(endY - h)

        # add the bounding box coordinates and probability score to the respective lists
        results.append((startX, startY, endX, endY))
        confidences.append(scoresData[x])

boxes = non_max_suppression(np.array(results), probs=confidences)

results = []

# loop over the bounding boxes
for (startX, startY, endX, endY) in boxes:
    startX = int(startX * rW)
    startY = int(startY * rH)
    endX = int(endX * rW)
    endY = int(endY * rH)

    dx = int((endX - startX) * padding)
    dy = int((endY - startY) * padding)

    # apply padding to each side of the bounding box respectively
    startX = max(0, startX - dx)
    startY = max(0, startY - dy)
    endX = min(imageW, endX + (dx * 2))
    endY = min(imageH, endY + (dy * 2))

    # extract the actual padded roi 'Region of Interest'
    roi = image[startY:endY, startX:endX]

    config = ("-l eng+kor --oem 1 --psm 7")
    text = pytesseract.image_to_string(roi, config=config)
    #text = pytesseract.image_to_string(roi, lang='eng+Hangul')
    # add the bounding box coordinates and OCR'd text to the list of results
    results.append((startX, startY, endX, endY, text))

results = sorted(results, key=lambda r:r[0][1])

# loop over the results
for ((startX, startY, endX, endY), text) in results:
    # display the text OOR'd by Tesseract
    print("OOR TEXT")
    print("=====")
    print("{}\n".format(text))

    # strip out non-ASCII text so we can draw the text on the image using OpenCV,
    # then draw the text and a bounding box surrounding the text region of the input image
    text = ''.join([o if ord(o) < 128 else ' ' for o in text]).strip()
    output = image.copy()
    cv2.rectangle(output, (startX, startY), (endX, endY), (0, 0, 255), 2)
    cv2.putText(output, text, (startX, startY - 5), cv2.FONT_HERSHEY_SIMPLEX, 1.2, (0, 0, 255), 3)

    # show the output image
    cv2.imshow("Text Detection", output)
    cv2.waitKey()

cv2.destroyAllWindows()
```

알약 식별 서비스 아키텍처 구조도





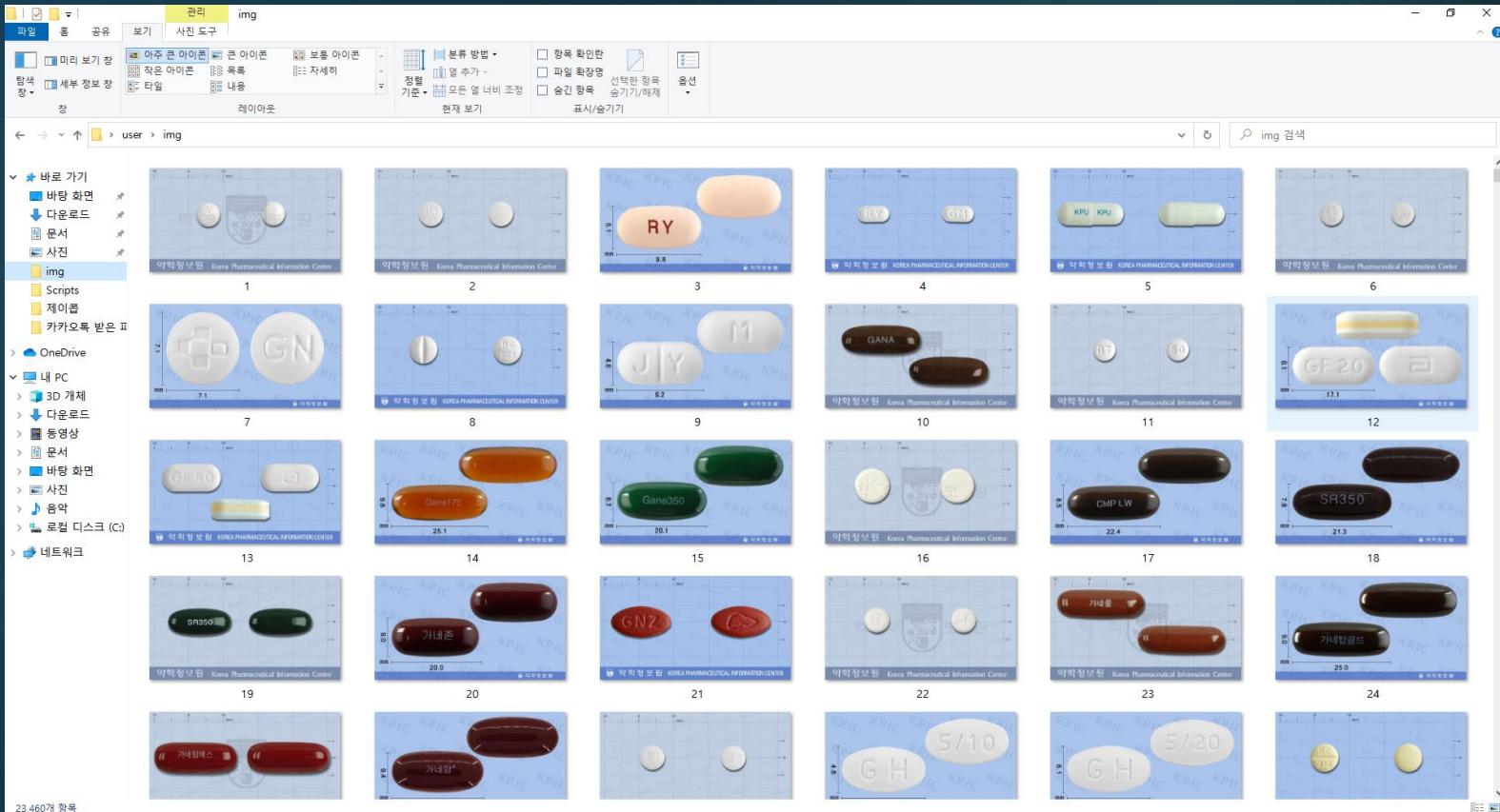
□ 아키텍처 설계

국내 최대

“52,609”

데이터 베이스 구축

크롤링 기법으로
약학정보원 정보 수집



▲ 크롤링 결과 이미지

데이터 베이스 구축

수집한 약학정보를
데이터베이스에 저장

주요 정보
제품명, 투여경로,
식약처분류 정보 추출

ITEM_NAME	ENTP_SEQ	ENTP_NAME	CHART
징카민정 40mg(은행엽건조엑스)(수출명 : 징카...)	19620008	영진약품(주)	(수출용) 하트형의 연녹색 제피정(내수용) 연녹...
아스코푸정(히벤즈산티페피딘)	19620008	영진약품(주)	황색의 정제임.
진셀몬정	19620008	영진약품(주)	장방형의 적갈색 제피정이다
저니스타서방정8밀리그램(히드로모르폰염산염)	19940315	(주)한국안센	빨간색의 원형, 양면이 볼록한 서방정
씨즈날정(세티리진염산염)	19910005	(주)녹십자	흰색의 타원형 필름코팅정
라니타드정	19910005	(주)녹십자	백색의 원형 필름코팅정
알레기살정10밀리그램(페미로라스트칼륨)	19660001	현대약품(주)	띠 모양 할선이 있는 황백색 원형정제이다
페나투신캡슐200밀리그램(구아이페네신)	19640003	미쓰비시다나베파마코리아주식회사	이 약은 백색분말로 총전된 상(불투명암녹색), ...
로부펜정(록소프로펜나트륨수화물)	19600001	삼아제약(주)	흰색의 원형정제
메트그린에스알정500밀리그램(염산메트포르민)	19910005	(주)녹십자	백색의 양쪽이 볼록한 장방형의 필름코팅정
도네오연질캡슐(수출용)	19910005	(주)녹십자	암녹색 내용물이 든 녹색의 장방형 연질캡슐제
액시드캡슐150밀리그램(니자티딘)	19540008	(주)대웅제약	백색~회백색의 분말을 충진한 상부황색, 하부...
푸루나졸캡슐50밀리그램(플루코나졸)	19540008	(주)대웅제약	이 약은 백색의 분말이 들어있는 상부 청록색, ...
マイ드린캡슐	19910005	(주)녹십자	백색의 분말 및 결정의 혼합물이 충진된 상·하...
데놀정(비스무트시트르산염칼륨)	19910005	(주)녹십자	원형의 백색 필름 코팅정
드로피진정(레보드로프로피진)	19650002	코오롱제약(주)	백색의 원형 정제
우루사정 100밀리그램(우르소데옥시콜산)	19540008	(주)대웅제약	흰색의 원형 정제
베아제정	19540008	(주)대웅제약	연녹색의 장방형 필름코팅정제이다.
대웅아스코르бин산정1000밀리그램	19540008	(주)대웅제약	백색의 장방형정제
대웅심바스타틴정20밀리그램(심바스타틴)(수...)	19540008	(주)대웅제약	황갈색의 타원형의 필름코팅정제
리버글드에프연질캡슐	19540008	(주)대웅제약	황갈색의 유상 내용물을 함유한 적갈색의 장방...
대웅심바스타틴정40밀리그램(수출명:Starzoko...)	19540008	(주)대웅제약	황갈색의 타원형 필름코팅정제

▲ mysql 데이터 이관 결과

데이터 베이스 구축

AWS EC2 서버로
데이터베이스 연결

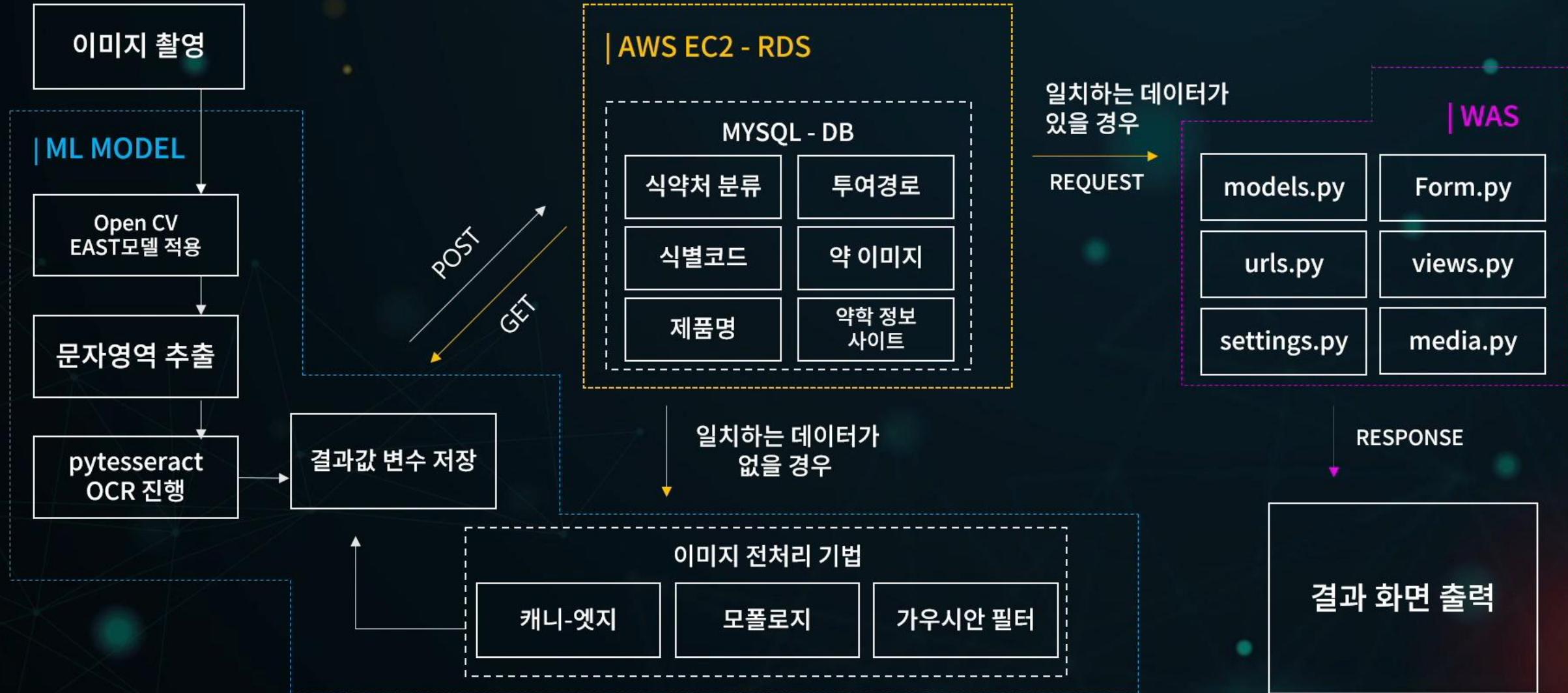
The screenshot shows the AWS EC2 Instances page. At the top, there is a blue header bar with the text "Welcome to the new Instances experience!" and a message about the console redesign. Below the header, the main title is "Instances (1) Info". A search bar labeled "Filter Instances" is present. The main table displays one instance:

Name	Instance ID	Instance state	Instance type	Status check	Alarm Status	Availability zone	Pub
-	i-0459d78b5b279e375	Running	t2.micro	Initializing	No alarms +	us-east-2c	ec2-

Below the table, a message says "Select an instance above". The left sidebar contains navigation links for EC2 Dashboard, Events, Tags, Limits, Instances (with sub-links for Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations), Images (AMIs), and Elastic Block Store (Volumes, Snapshots, Lifecycle Manager). The bottom of the page includes a feedback link, language selection (English (US)), and a footer with copyright information and links to Privacy Policy and Terms of Use.

▲ AWS-EC2 인스턴스 생성 결과

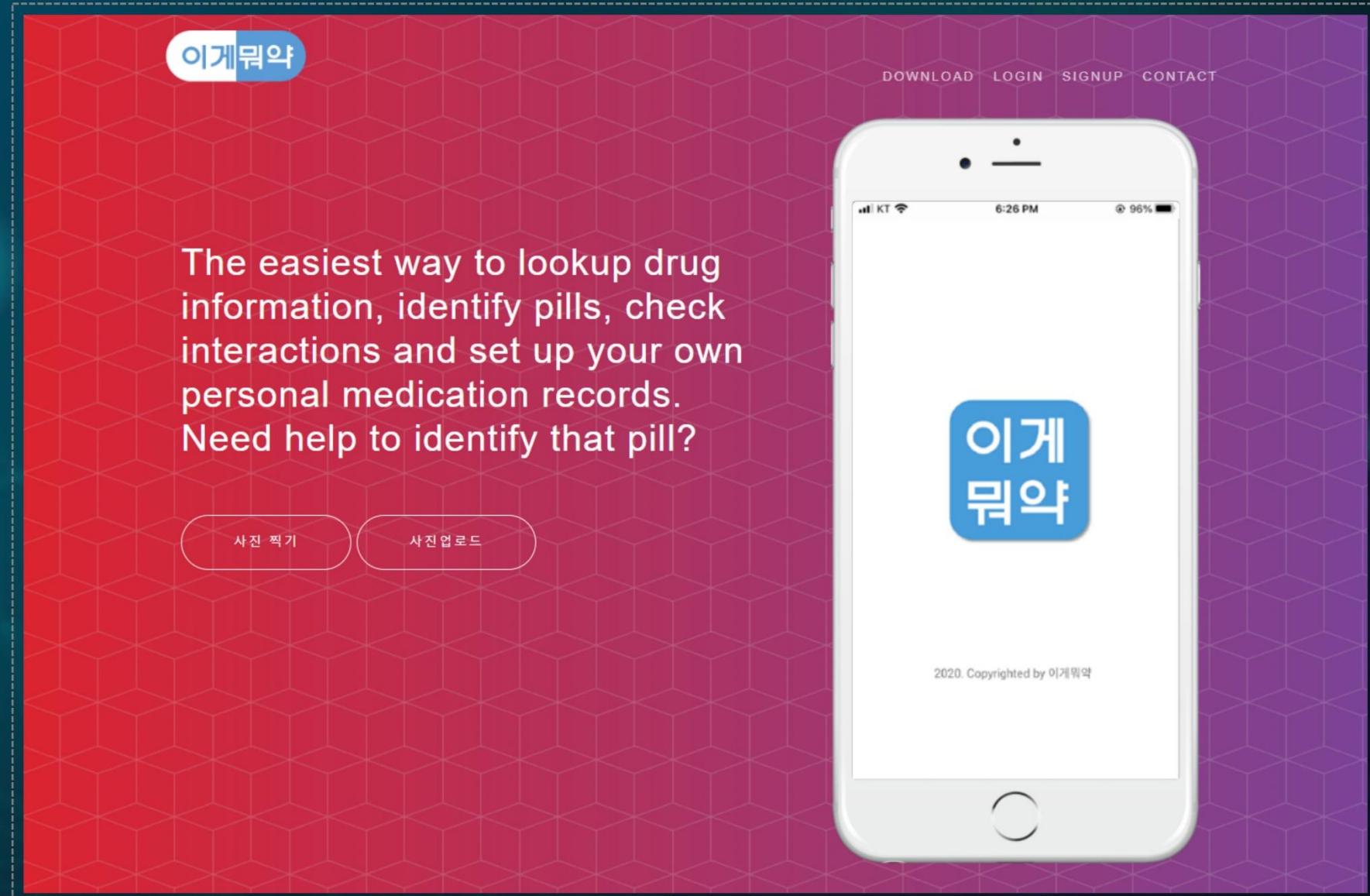
알약 식별 서비스 아키텍처 구조도



강력한
도구를
소개합니다

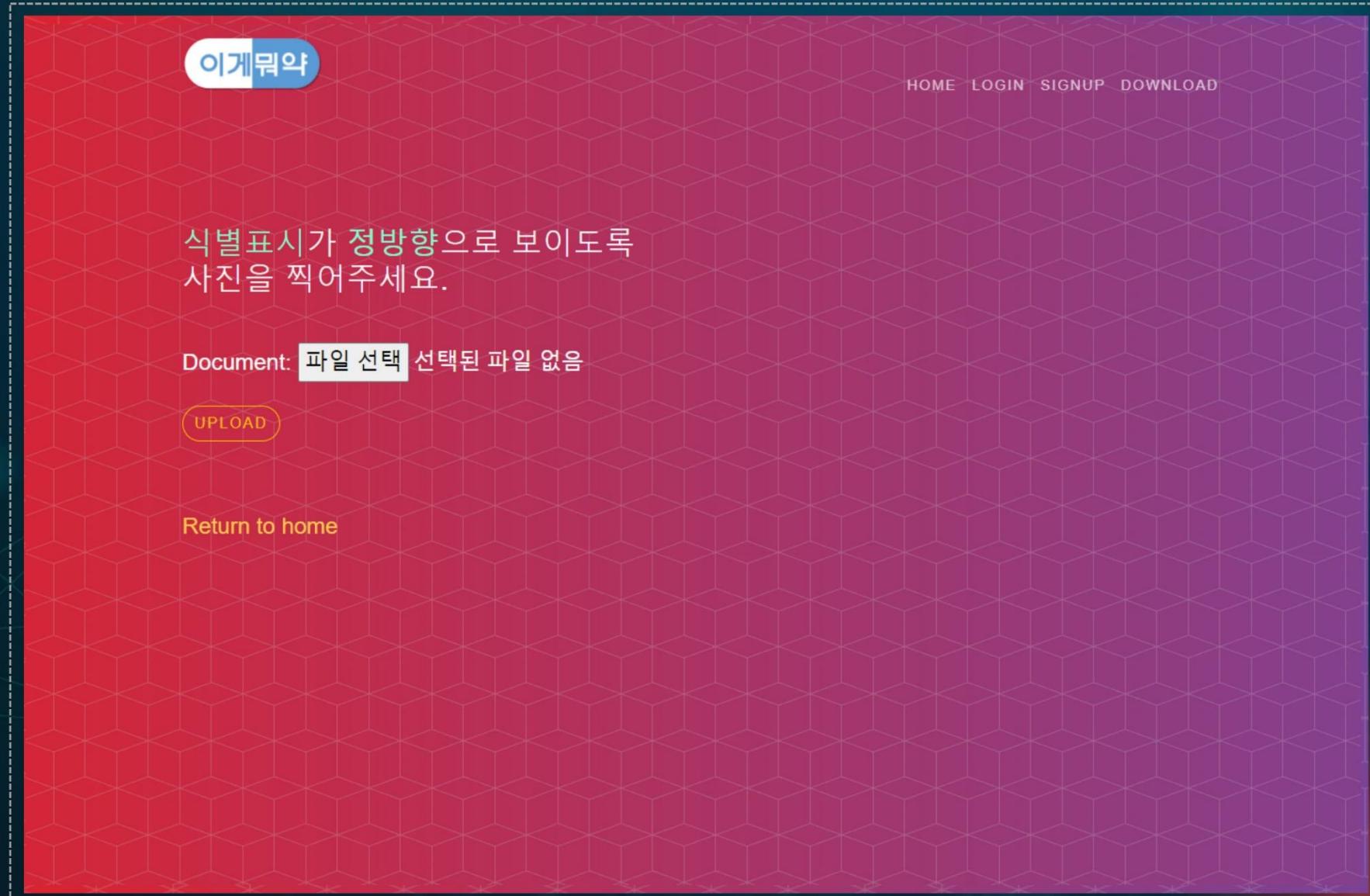
WEB

웹사이트 구현 모습



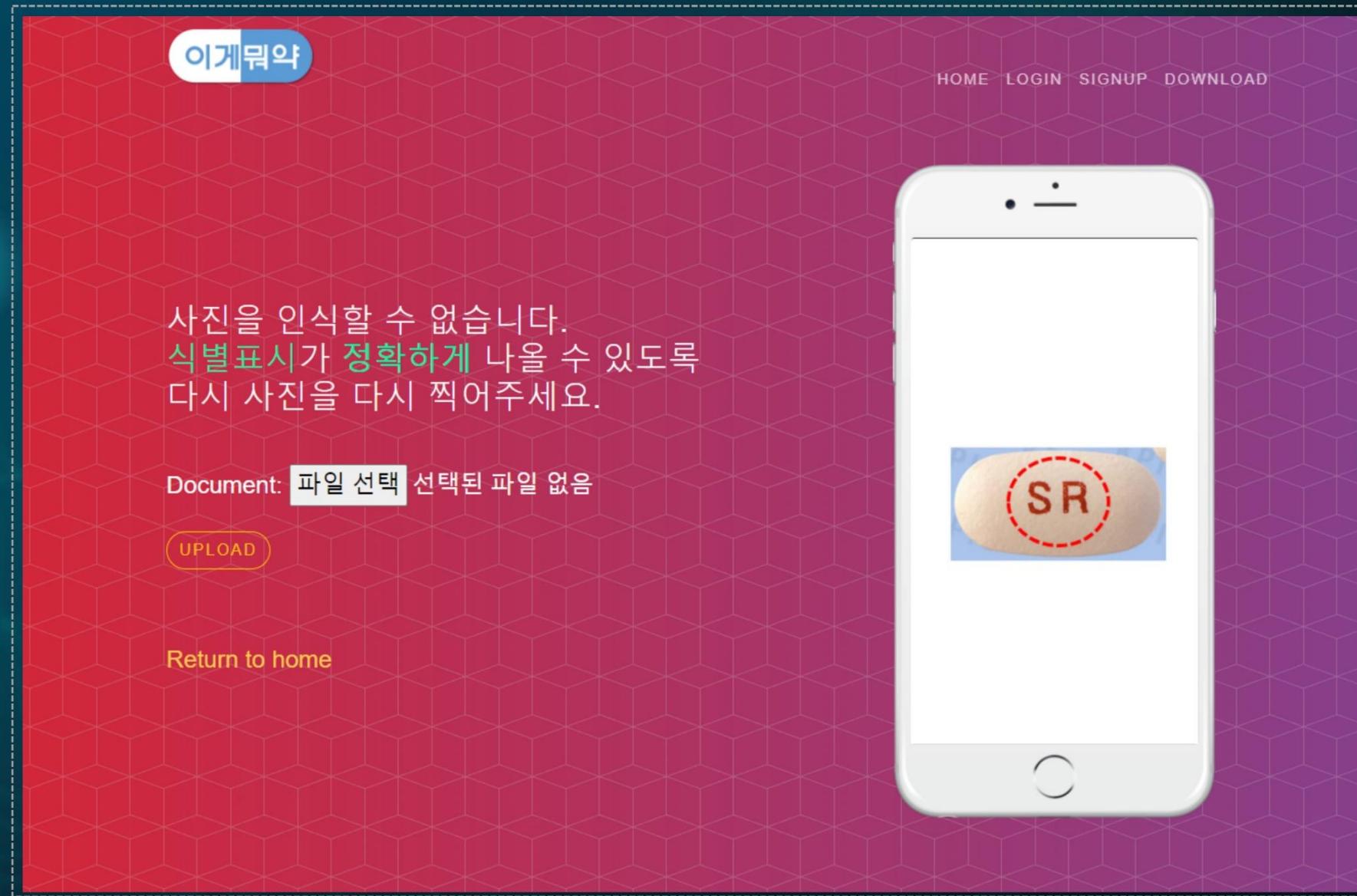
WEB

약 검색 페이지



WEB

인식 실패 안내



WEB

결과 출력 화면



APP

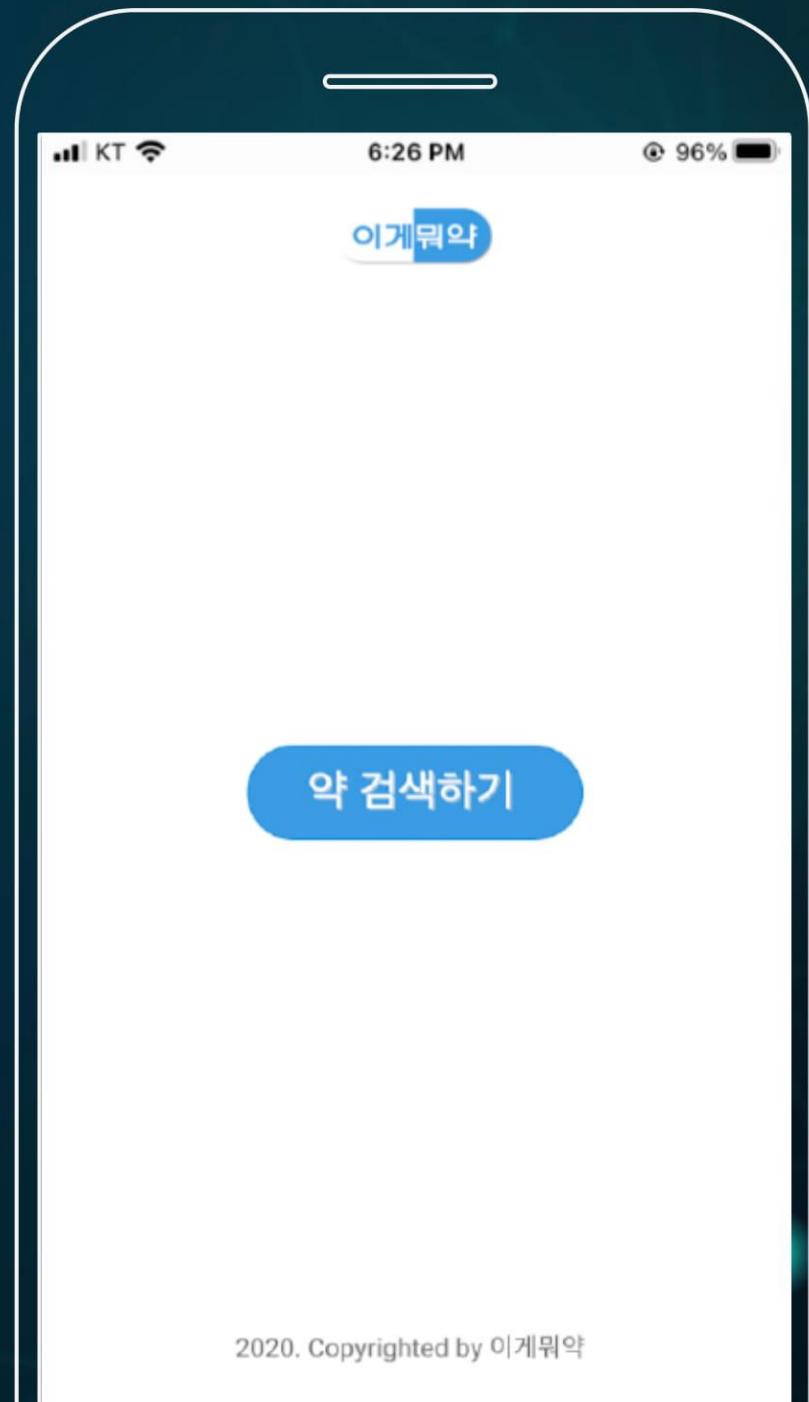
어플리케이션 메인 화면

이게
뭐약

2020. Copyrighted by 이게뭐약

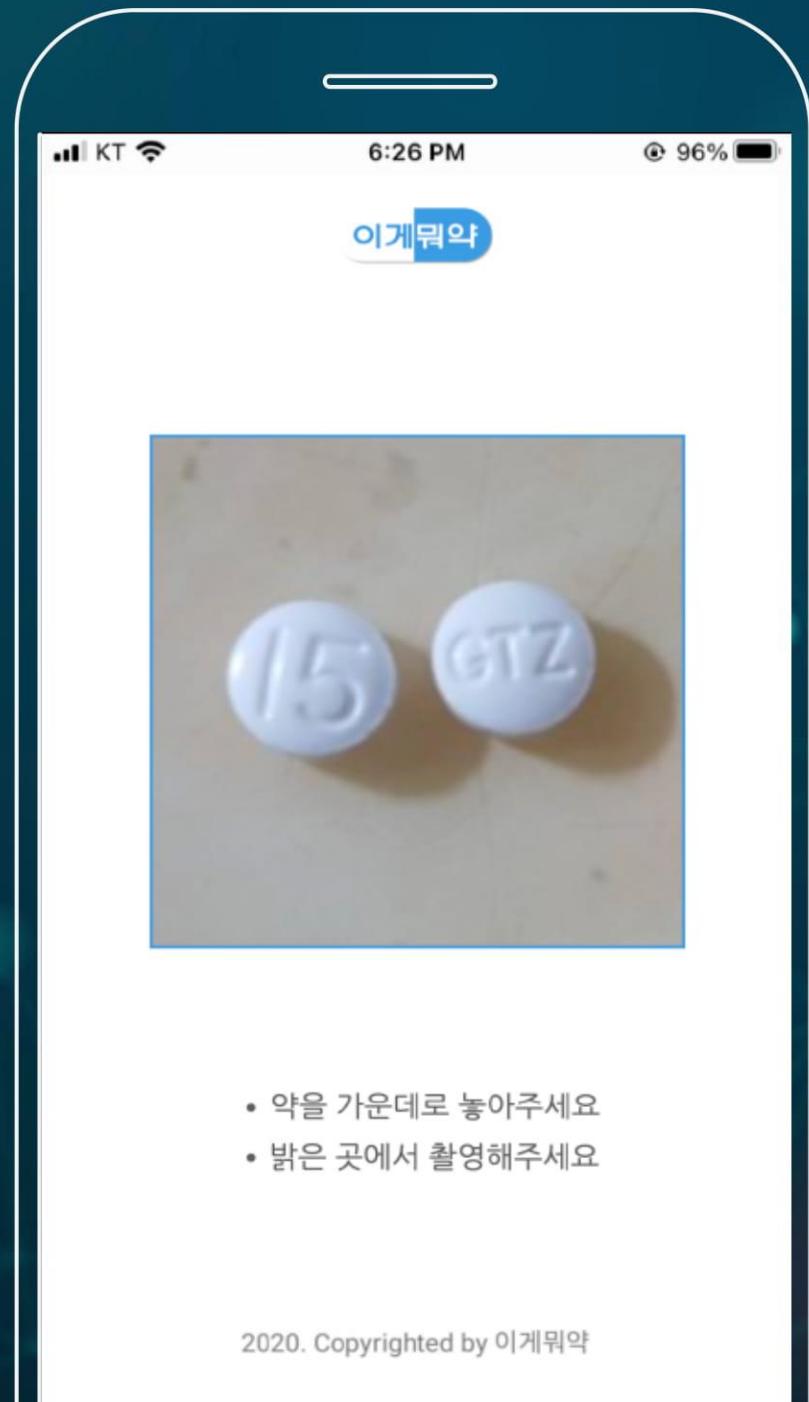
APP

자동으로 약 검색 유도



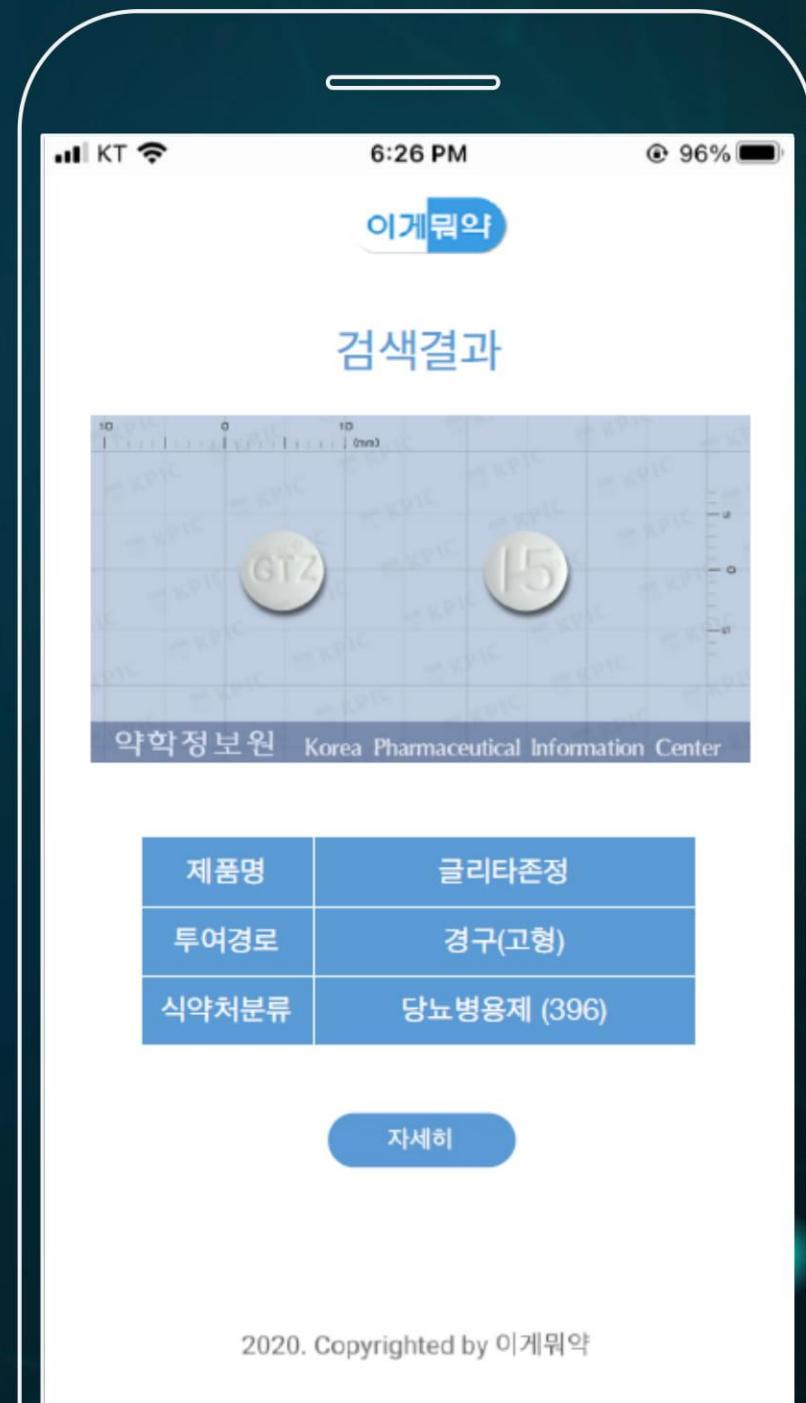
APP

사진만 찍으면 완성되는
친화적인 사용자 UI



APP

사용자에게 필요한 핵심 정보만 전달



시연 04

Demonstration part

결론 05

결론 및 개선점

정량적 효과

한 해, 폐기되는 약을 비용으로
환산했을 경우 **2,180억 원** 가량

중복 처방으로 버려지는 약의 비율

63.4%



이게뭐약 서비스를 통해

약 **1,382**억 원
절약 가능

이게뭐약 서비스를 통해

약 **2명** /3명 중



정보 접근성 ↑

약 **1명** /2명 중



약화 사고 발생 ↓

정량적 효과 II

대한 약사회에서 서울 거주 시민
192명에게 물었다.

63.4%

Q. 가정에서 보관 중인 약의 용법을 아시나요?
A. 잘 모르겠습니다.

47.8%

Q. 보관중인 의약품을 정확한 정보 없이
대강의 짐작으로 복용한 적이 있나요?
A. 네 있습니다.

정성적 효과 및 개선점



1. 플랫폼 개선으로 사용자 편의성 향상 기대



2. 음각 이미지 알고리즘 개선 가능



3. 가능 먼저, 디자인은 천천히

“If we follow
the “do something” principle,
failure feels unimportant.

When the standard of success becomes
merely acting ...
we propel ourselves ahead.”

‘무언가를 해내야 한다’면,
실패는 전혀 중요해보이지 않을 수 있다.
하지만, ’뭐든 해 보자’가 성공의 기준이 된다면
우리는 한걸음 더 성장할 수 있을 것이다.

모두 수고하셨습니다.

THE
END