

M.A.R.S. 2025 SNUBH 의무기록 생성 데이터톤

Medical Auto-documentation with Real-world Structuring:
LLM Clinical Note Generation Challenge

본선 결과 보고서 [MediX]

주최/주관



후원



0. 팀 소개 (MediX)

팀 개요 및 역할 분담 : 의료AI의 혁명을 꿈꾸는 학부 3학년 개인 참가자입니다!

ABOUT ME

Hi, I'm Seoyoung Oh. I have a strong interest in Healthcare and Medical AI, and I aspire to drive innovation in the medical domain through the power of data and artificial intelligence. Among the many fields of AI and data science, I find healthcare to be the most meaningful and impactful, as it directly relates to people's lives and well-being. My academic focus is centered on exploring how AI and data-driven solutions can address real world challenges in medicine, and I'm actively seeking opportunities to grow through research, interdisciplinary collaboration, and practical applications in this field.

수상 실적

- [2023] 국민대학교 기후변화대응 비즈니스 아이디어 공모전 장려상
- [2024] 제2회 국민대학교 AI빅데이터 분석 경진대회 우수상
- [2025] 국민대학교 경영대학 창업경진대회 최우수상
- [2025] 춘천시 생성형 AI 아이디어 경진대회 우수상
- [2025] 제9회 디지털 헬스케어 MEDICAL HACK 최우수상 (부산대학교총장상)
- [2025] 건양대학교의료원 KHD (Konyang Health Datathon) 최우수상
- [2025] 국민대학교 개교 79주년 기념 학생 포상 국민인재상 (총장 표창장)

창업 실적 | TEAM. MEDIX (대표 AI 개발자)

- [2025] 서울 AI 허브 - Seoul AI Young Track (SAY 트랙) | 최종 선정 및 수료
- [2025] 연세대학교 Y-Compass 학생창업팀 | 최종 선정 (A등급 수료)
- [2025] 부산대병원 주최 - 제 9회 디지털 헬스케어 MEDICAL HACK 2025 | 최우수상
- [2025] 학생창업유망팀 U 300+ | 최종 선정 및 활동 중

주요 프로젝트

- [2024] 카카오톡 챗봇 기반 인지증재치료 서비스 제작
- [2025] AI 기반 임신 주수별 개인 맞춤 약물 섭취 금기 모니터링 서비스 제작
- [2025] EMR 기반 의료진 간의 인수인계를 위한 기반 변동사항 요약 시스템 제작 → 창업팀 활동
- [2025] 측면두부규격방사선 분석 기반 부정교합 진단 및 랜드마크 자동 검출 AI 모델 제작 → KHD 수상 프로젝트
- [2025] An Explainable Framework for Diabetic Retinopathy Classification Using Fundus Image Preprocessing and Visual Explanations → 2025 대한의료인공지능학회 정기학술대회 포스터 채택
- [2025] PPG 기반 호흡수 추정의 신뢰도 향상 SQI-SNR 게이팅과 Conformal 예측구간 → 2025 대한의료정보학회 추계학술대회 포스터 채택
- [2025] Surgical Skill Assessment Using Kinematic Features and Deep Learning on the JIGSAWS Dataset → 2025 의료메타버스학회 추계학술대회 포스터 채택
- [2025] Machine Learning Models for Predicting Operative Duration in Assisted Reproductive → BMC MEDICAL RESEARCH METHODOLOGY 논문 투고 진행 중
- [2025] Korea Clinical Datathon 2025 참가

CONTACT

E-mail : 5seo0_oh@kookmin.ac.kr
Linkedin: <https://www.linkedin.com/in/5seo0/>
GitHub : <https://github.com/5seoyoung>
Blog : <https://5seo0.tistory.com/>

학력

- 안양외국어고등학교 졸업 (2020.03 ~ 2023.02)
- 국민대학교 AI빅데이터융합경영학과 재학(2023.03 ~)
- 국민대학교 인공지능학부 부전공(2025.03 ~)

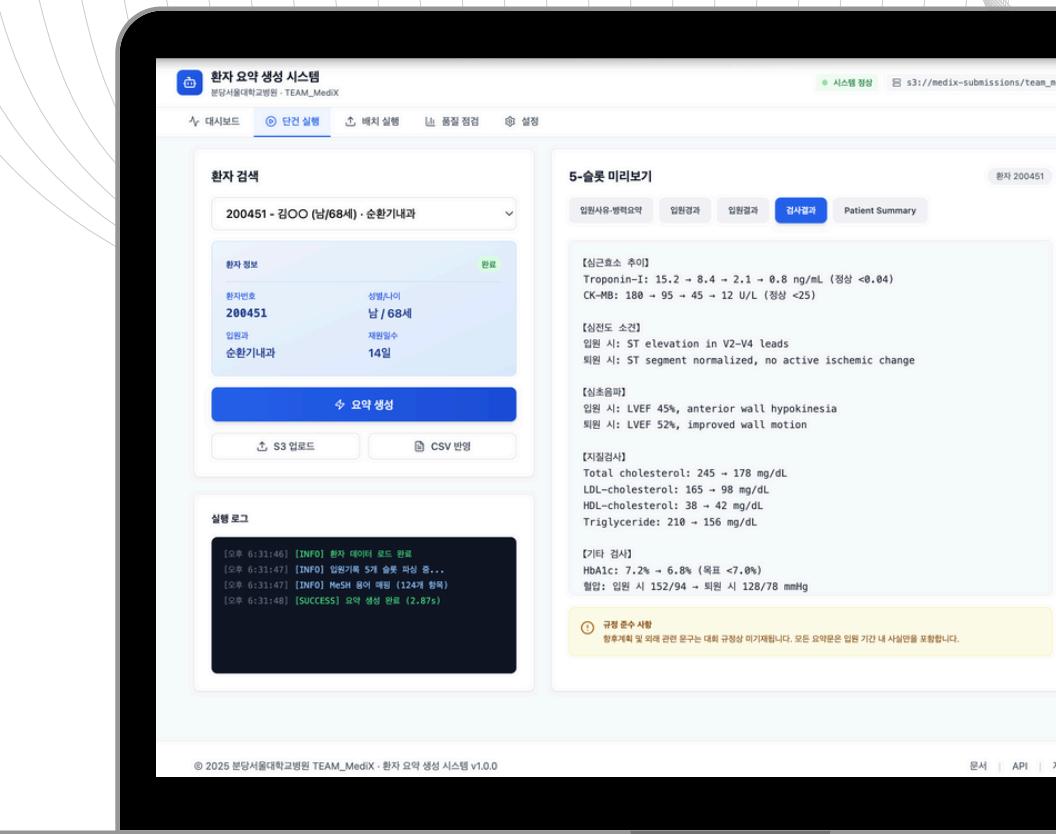
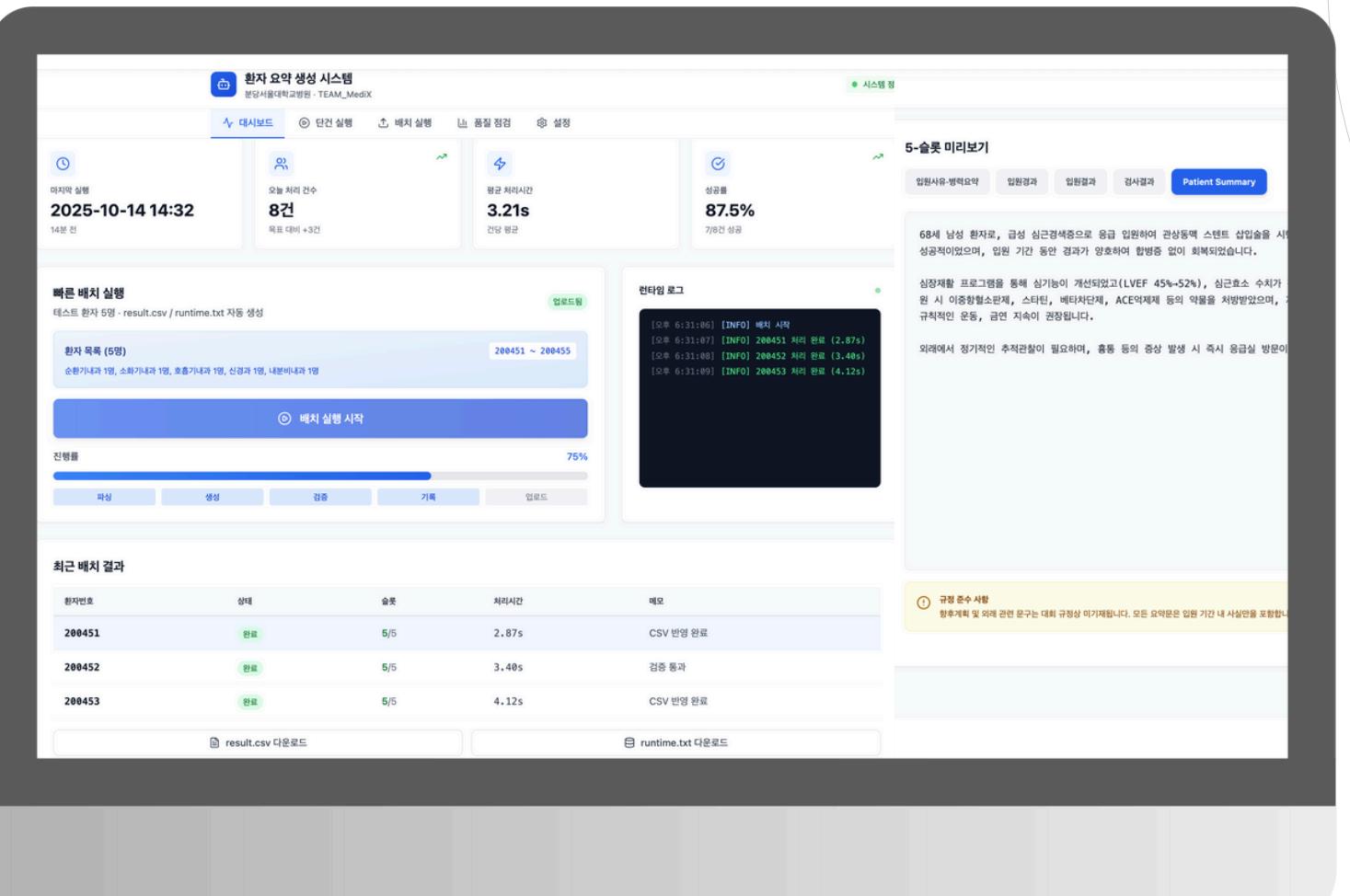
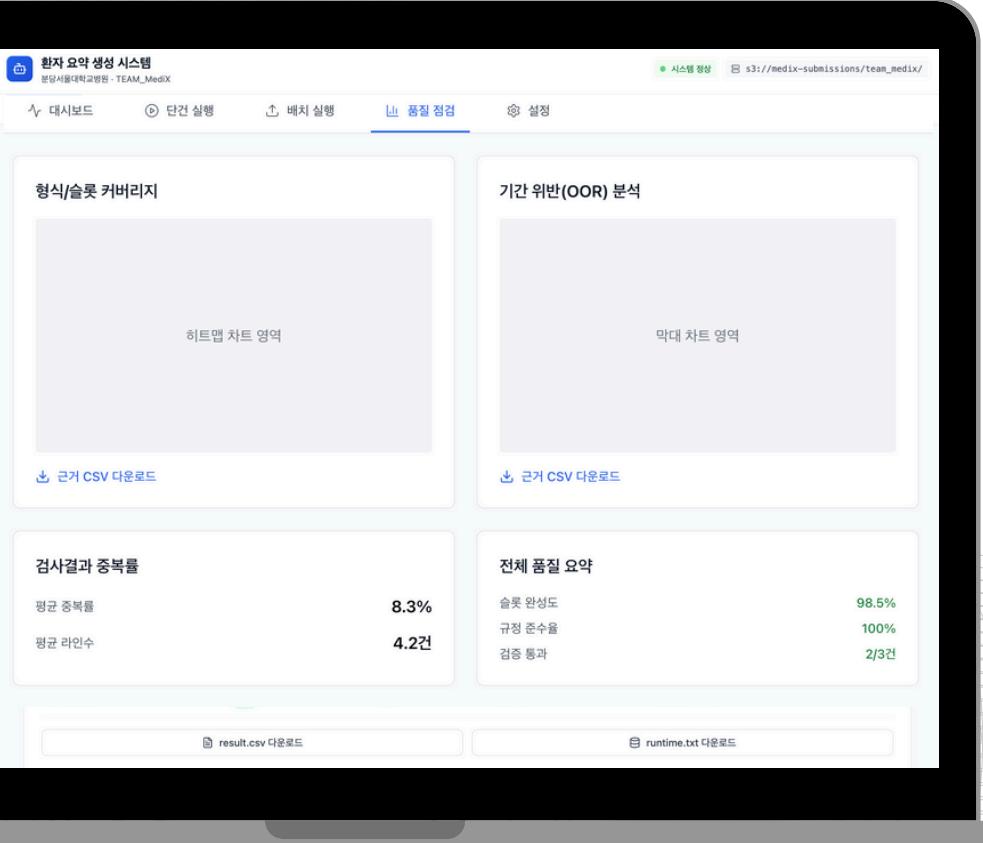
개인 참여이지만, 팀처럼 체계적으로!

- 프롬프트 설계 → Task별 프롬프트 디자인 및 후처리 규칙 개발
- 데이터 분석 → Train 분포 분석, 시각화 및 근거 제공
- 보고서/발표 → 전략·실험결과를 정리하고 임상적 의미 해석

M.A.R.S. 2025 SNUBH 의무기록 생성 데이터톤

Medical Auto-documentation with Real-world Structuring:
LLM Clinical Note Generation Challenge

입원~퇴원 구간의 임상기록에서 규칙·정규식 + MeSH 정규화로 핵심 정보를 뽑아
LLM 없이도 결정적으로 필수 5슬롯 퇴원요약을 자동 생성하는 파이프라인



1. 문제 정의 & 접근 개요

1.1 정량적 목표와 성공 판단 기준

요구사항



입력(6개 테이블)



제약(퇴원기록 미사용·기간 필터·향후계획 미기재)



산출물(result.csv, runtime.txt)

|

슬롯 커버리지 100%

범위 밖 레코드 ≈ 0

동일 Lab 항목 표기 ≤ 2줄

8건 총 시간 XX.s, 건당 평균±표준편차

최종 목표

환자별로 가이드의 ★필수 5슬롯을 모두 채운 퇴원요약 1건을 코드만으로 자동 생성
품질(형식·정확성·가독성)과 효율(처리시간) 동시 달성. 운영은 비공개 8명 테스트셋으로 실행하며 runtime.txt의 총 소요 시간을 평가에 반영

문서 구조(핵심 슬롯)

입원사유·병력요약(★), 입원경과(★), 입원결과(★), 검사결과(★), Patient Summary(★)—정해진 이름/순서 유지

데이터 사용 규칙

퇴원기록.csv 입력 금지(참고자료일 뿐)
퇴원일 이후 기록 사용 금지, 결측 대비 예외처리 필수
모든 섹션이 동일한 기간 필터(within_stay)를 사용 → 시간적 일관성 보장

1. 문제 정의 & 접근 개요

1.2 평가 기준 ↔ 코드 매핑표

형식/슬롯 → TEMPLATE_5 format_with_template_5

```
def within_stay(df, pid, id_col, date_col, t_in, t_out):
    """입원≤date≤퇴원 구간 필터 + 정렬."""
    dd = df.loc[df[id_col] == pid].copy()
    dd[date_col] = to_dt(dd[date_col])
    ok = dd[date_col].notna() & (dd[date_col] >= t_in) & (dd[date_col] <= t_out)
    return dd.loc[ok].sort_values(date_col)
```

정확성(날짜/수치) → within_stay _NUM_PAT

일관성 → 모든 섹션 within_stay + post_kor

→ 평가표를 ‘함수 포인트’로 매핑했습니다

유용성 → enforce_specialty_points 기본 OFF

운영 평가표의 각 항목을 함수 레벨로 직접 매핑

간결성 → MAX_TOTAL_CHARS & 동일 항목 ≤2

수치 파서는 단위/동의어까지 처리하고, 기간 경계는 모든 테이블에 공통 적용

간결성은 하드 컷(문자수) + 중복 억제(최신 2개)로 제어

효율성 → run_submit_batch write_runtime

1. 문제 정의 & 접근 개요

1.3 문서 구조와 상위 KCD 진단 TOP-N 분석

```
# 8) 5-슬롯 템플릿/생성 -----
REQUIRED_SLOTS = [
    "admission_reason_history",
    "hospital_course",
    "outcome",
    "key_results",
    "patient_summary"
]

TEMPLATE_5 = """[퇴원요약]
입원사유·병력요약: {admission_reason_history}
입원경과: {hospital_course}
입원결과: {outcome}
검사결과:
{key_results}
Patient Summary:
{patient_summary}
""".strip()
```



```
def _fill_required_5(sections: dict) -> dict:
    filled = dict(sections)
    for k in REQUIRED_SLOTS:
        if not filled.get(k):
            filled[k] = "해당 항목 기재 없음"
    return filled
```

누락 시 ‘해당 항목 기재 없음’ 자동 보정

이름/순서 1:1 고정

이름/순서 1:1 고정(평가 스키마 일치)

슬롯 누락 시 “해당 항목 기재 없음” 자동 보정

Patient Summary는 앞 슬롯을 5~7문장으로 응축

	KCD 한글명	count
1	기타 바이러스질환에 대한 특수선별검사	50
2	상세불명의 심부전	50
3	상세불명의 급성 신부전	42
4	만성 신장병(5기)	41
5	파열되지 않은 대뇌동맥류	36
6	담관염	35
7	상세불명의 체장의 악성 신생물	35
8	상세불명의 급성 심근경색증	34
9	상세불명의 심방세동	30
10	죽상경화성 심장병	28
11	흑색변	28
12	상세불명의 만성 신장병	25
13	급성 심내막하심근경색증	23
14	급성 세뇨관-간질신장염	20
15	간외담관의 악성 신생물	19

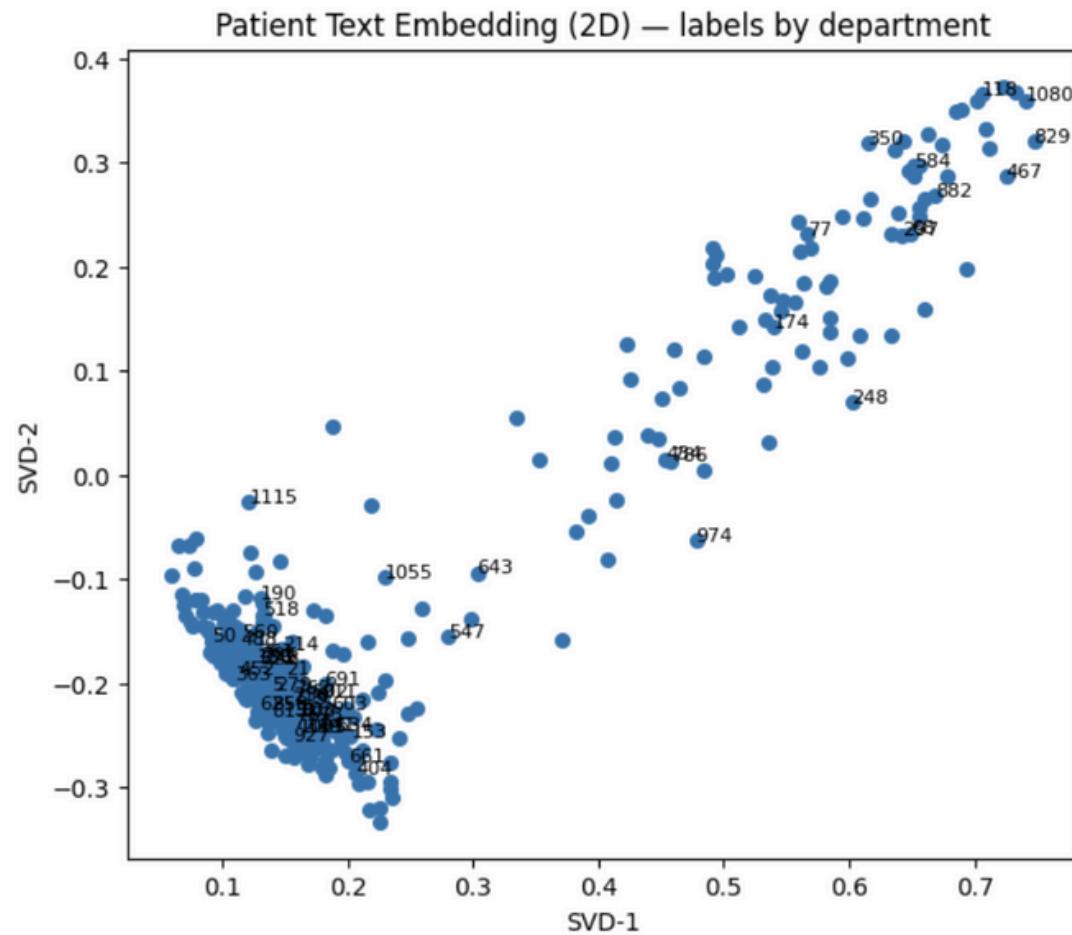


상위 5개 진단이 전체의 다수를 차지

신장/소화기/심장계 비중 높음 → 해당 계열 용어·검사(예: CR/EGFR, EF, 내시경) 우선 반영
질환 스펙트럼이 넓어 동의어 규칙과 정규화가 필수 → MESH 매핑·라벨 정규화 채택

1. 문제 정의 & 접근 개요

1.4 환자 임상텍스트 분석 결과



환자 임상텍스트를 2차원으로 임베딩(SVD-1, SVD-2 축) 해, 진료과별 레이블로 표시한 산점도

- 점 하나 = 환자 1명의 텍스트(입원경과 등)를 벡터화→차원축소해 배치한 위치
- 축(SVD-1, SVD-2)는 텍스트 의미공간의 주성분 좌표로, 값 자체의 해석보다는 상대적 거리/군집이 중요
- 가까운 점들 = 유사한 문체·키워드(질환 스펙트럼/검사·시술 용어 등)
- 좌하단에 내과계(내분비·소화기·종합내과 등) 가 주로 군집, 우상단에 심혈관/외과계가 군집하여 과별 문서 패턴이 뚜렷이 분리 됨을 보여줌
- 하단 표의 DEPT CENTROIDS(2D)는 각 과의 클러스터 중심 좌표, 즉 대표적인 문서 패턴의 위치를 요약한 값

→ 과별로 문체와 핵심 키워드가 달라 전문화된 규칙(예: EF/ECG/투석/내시경 등)을 별도로 적용하면

요약 품질을 높일 수 있다는 근거

2. ‘3-Stage’ Architecture

2.1 공통 세팅: 데이터·경계·출력 사양

테이블



기간 필터



슬롯 빌더



산출물(result.csv, runtime.txt)

```
# Stay & time filter -----
def get_stay(pid: int):
    """환자별 입원~퇴원 구간 반환."""
    rows = admission.loc[admission["환자번호"] == pid]
    if rows.empty:
        raise ValueError(f"환자번호 {pid} 없음")
    row = rows.iloc[0]
    t_in = to_dt(row["입원일자"])
    t_out = to_dt(row["퇴원일자"])
    if pd.isna(t_in) or pd.isna(t_out):
        raise ValueError(f"환자번호 {pid} 입원/퇴원일 결측")
    return t_in, t_out

def within_stay(df, pid, id_col, date_col, t_in, t_out):
    """입원≤date≤퇴원 구간 필터 + 정렬."""
    dd = df.loc[df[id_col] == pid].copy()
    dd[date_col] = to_dt(dd[date_col])
    ok = dd[date_col].notna() & (dd[date_col] >= t_in) & (dd[date_col] <= t_out)
    return dd.loc[ok].sort_values(date_col)
```

Stage A



사실 수집·정제(정규식/키워드/기간 필터)

Stage B



문장화(규칙 기반, LLM OFF 기본)

Stage C

GUARDRAILS & EXPORT(형식/사실/출력)

→ 입원≤T≤퇴원, 퇴원기록.CSV 미사용, 출력 포맷 고정

기간 밖 데이터는 전 섹션에서 공통으로 제외됩니다

2. ‘3-Stage’ Architecture

2.1 공통 세팅: 데이터·경계·출력 사양

Stage A

사실 수집·정제(정규식/키워드/기간 필터)



Stage B

문장화(규칙 기반, LLM OFF 기본)



Stage C

GUARDRAILS & EXPORT(형식/사실/출력)

	Area	Functions
1	기간 필터	get_stay, within_stay
2	5슬롯 템플릿	REQUIRED_SLOTS, TEMPLATE_5, format_with_template_5
3	경과 요약	make_med_course, KEY_NOTE_FIELDS
4	검사 추출	_NUM_PAT, _canonize, extract_key_results
5	MeSH 통합	_compile_hint_from_mesh, _load_mesh_lab_alias, _canonize
6	과별 힌트	guess_specialty, enforce_specialty_points
7	입·출력	run_submit, run_submit_batch, write_runtime

→ 규칙 기반 파이프라인 구조

시간 구간 필터로 입원≤T≤퇴원만 사용(규정 준수)

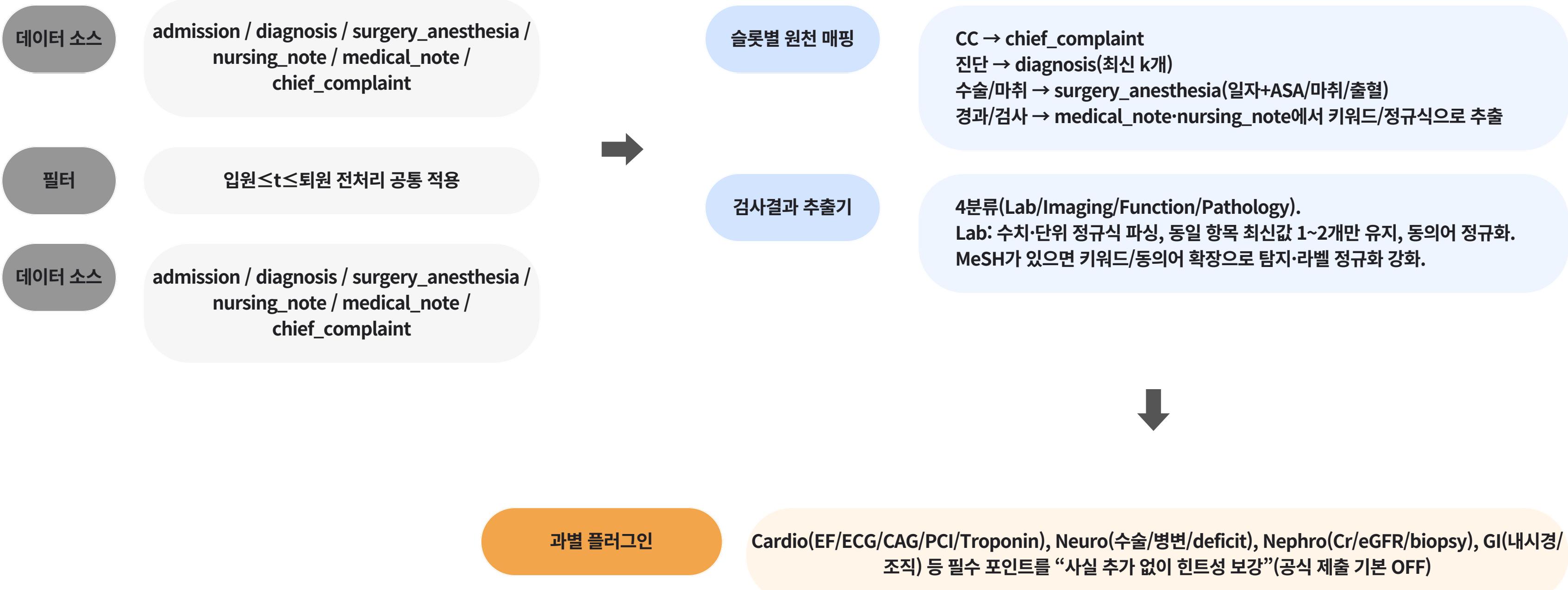
5-슬롯 템플릿으로 형식 강제

검사/영상/병리 정규표현식 + MESH 동의어

LLM OFF, 결정적·재현 가능 아키텍처

2. ‘3-Stage’ Architecture

2.2 STAGE A - 사실 수집·정제 플로우



2. ‘3-Stage’ Architecture

2.3 STAGE A 세부 - 경과·진단·수술

```
# 경과 추출 키워드
KEY_NOTE_FIELDS = re.compile(r"(Assessment|Impression|Plan|진단|경과|요약|소견)", re.I)

def make_med_course(pid, t_in, t_out, max_chars=800):
    """입원경과 요약"""
    # Medical notes
    m = within_stay(medical_note, pid, "환자번호", "서식 작성일자", t_in, t_out)
    m_lines = []
    for _, r in m.iterrows():
        name = s(r.get("서식 항목명"))
        val = s(r.get("항목별 서식 값"))
        if not val:
            continue
        if KEY_NOTE_FIELDS.search(name) or KEY_NOTE_FIELDS.search(val):
            m_lines.append(f"{name}: {val}")
    # Nursing notes
    n = within_stay(nursing_note, pid, "환자번호", "간호기록 작성일자", t_in, t_out)
    n_key = re.compile(r"(내원 주증상|입원동기|의식|활력징후|배변|심취|통증|산소|증상|간호)", re.I)
    n_lines = []
    for _, r in n.iterrows():
        key = s(r.get("항목명"))
        val = s(r.get("항목값"))
        if n_key.search(key) and val:
            n_lines.append(f"{key}: {val}")
    return join_lines(m_lines + n_lines, max_chars=max_chars)

# 5) 검사결과 추출(기본 정규식/라벨)
_NUM_PAT = re.compile(
    r"(?P<name>(Hb|Hemoglobin|헤모글로빈|WBC|Cr|Creatinine|크레아티닌|BUN|Na|K|칼륨|Sodium|ALT|AST|CRP|Troponin|EF|eGFR|Platelet|혈소판) [^:\n]{0,30})"
    r"[:\s]*"
    r"(?P<val>[+]?d+(?:\.\d+)?)\s*(?P<unit>%|mg/dL|g/dL|U/L|mmol/L|mEq/L|x10^-3/?u?L|ng/L|mL/min/1|.73m^2)?",
    re.I
)
IMG_HINT = re.compile(r"(CT|MRI|X[- ]?ray|Ultrasound|Echo|Echocardiography|angiography|내시경|초음파|흉부단순|CAG|PCI)", re.I)
FUNC_HINT = re.compile(r"(PFT|ABGA|EKG|ECG|EEG|Holter|TTE|TEE|EF)", re.I)
PATH_HINT = re.compile(r"(biopsy|조직검사|세포검사|병리|pathology)", re.I)

_SYNONYM_CANON = [
    (re.compile(r"creatinine[^a-z]cr[a-z]\b", re.I), "Creatinine (Cr)"),
    (re.compile(r"hemoglobin|hb|헤모글로빈", re.I), "Hemoglobin (Hb)"),
    (re.compile(r"\bwbc\b", re.I), "WBC"),
    (re.compile(r"platelet|혈소판", re.I), "Platelet"),
    (re.compile(r"egfr", re.I), "eGFR"),
    (re.compile(r"troponin", re.I), "Troponin"),
    (re.compile(r"\bna\b|sodium", re.I), "Sodium (Na)"),
    (re.compile(r"\bk\b|칼륨", re.I), "Potassium (K)"),
]

```

→ 핵심 필드 큐레이션 + 날짜 중심

길이 제한으로 노이즈를 제거하고 가독성을 확보

환자번호	입원시 진료과	문서라인수	총문자수	총단어수(공백기준)	코퍼스
0	신장내과	16	5235	775	Assessment Current>\n\n#. AKI R/O pre-reenal (F...
1	순환기내과 (심장혈관센터)	8	2180	393	normal <- General condition <- Reportable Find...
2	순환기내과 (심장혈관센터)	42	5020	854	수신의1 <- 병원공통_타과의뢰_기본서식 OOO \n normal <- Overall...
3	순환기내과 (심장혈관센터)	94	15614	2791	Reportable Findings <- ROS @ Chest AP (2024-11...
4	신경외과 (뇌신경센터)	59	4596	961	의뢰의 <- 병원공통_타과의뢰_기본서식 서지민 \n Present illness 2...

환자별 코퍼스 샘플 표

원문 코퍼스 품질 점검(샘플)

서식명/항목명 기반으로 ‘경과/요약/소견’ 핵심 문장 우선 추출

퇴원 서식 라인은 런타임에서 배제(규정 준수)

잡음 최소화 → 의미 밀도 높은 입력 확보

2. ‘3-Stage’ Architecture

2.4 STAGE A 세부 - 검사결과

```
# 6) 검사결과 추출 -----
def extract_key_results(pid, t_in, t_out, max_lines=12) -> str:
    """
    검사결과(★): Lab/Imaging/Function/Pathology 4블록
    - Lab: 수치/단위 정규식 파싱 + 동의어 정규화 + 동일 항목 최신값 최대 2개 선택
    - Imaging/Function/Pathology: (MeSH 통합) 힌트 정규식 기반 탐지, 첫 문장 발췌 + 날짜
    """
    m = within_stay(medical_note, pid, "환자번호", "서식 작성일자", t_in, t_out)
    n = within_stay(nursing_note, pid, "환자번호", "간호기록 작성일자", t_in, t_out)
    rows = (
        _scan_note_rows(m, "서식 작성일자", ["서식 항목명", "항목별 서식 값"]) +
        _scan_note_rows(n, "간호기록 작성일자", ["항목명", "항목값"])
    )

    labs_map = {} # canon -> list[(date, [d] name: valunit")]
    imgs, funcs, paths = [], [], []

    for d, txt in rows:
        dtags = f"[{d.date()}]" if pd.notna(d) else ""
        # Lab 수치
        for mobj in _NUM_PAT.finditer(txt):
            raw_name = re.sub(r"\s+", " ", mobj.group("name")).strip(":")
            canon = _canonize(raw_name)
            val = mobj.group("val")
            unit = mobj.group("unit") or ""
            item_txt = f"{dtags} {canon}: {val}{unit}".strip()
            labs_map.setdefault(canon, []).append((d, item_txt))
    # Imaging / Function / Pathology (첫 문장)
    snippet = re.split(r"\n", txt)[0].strip()
    if IMG_HINT.search(snippet):
        imgs.append(f"{dtags} {snippet}")
    if FUNC_HINT.search(snippet):
        funcs.append(f"{dtags} {snippet}")
    if PATH_HINT.search(snippet):
        paths.append(f"{dtags} {snippet}")

    return "\n".join(labs_map.get(canon, []) for canon in sorted(labs_map))
```

→ LAB/IMAGING/FUNCTION/PATHOLOGY로 분류하여 날짜
라벨과 함께 최신값 1~2개만 채택

동의어→정규 라벨 매핑으로 중복/혼용 제거

	alias	canonical
1	Cr	Creatinine (Cr)
2	Creatinine	Creatinine (Cr)
3	크레아티닌	Creatinine (Cr)
4	Hb	Hemoglobin (Hb)
5	헤모글로빈	Hemoglobin (Hb)
6	Na	Sodium (Na)
7	K	Potassium (K)
8	혈소판	Platelet

LAB ALIAS → CANONICAL 매핑 표

검사 항목 정규화(동의어→정규 라벨)

Cr/크레아티닌 → Creatinine (Cr)

Hb/헤모글로빈 → Hemoglobin (Hb)

Na/K/혈소판 등 오탈자·혼용 흡수

→ 라벨 정규화로 중복 제거·요약 품질 상승

의미 중심 요약: 동일 지표 다중 표기를 한 줄로 통합

라벨 정규화로 잡음 ↓, 요약 품질 ↑

2. ‘3-Stage’ Architecture

2.5 STAGE B - LLM 요약·문장화 (옵션, 기본 OFF)

```
# 2) Settings & Utilities -----
TEST_DIR    = Path("test")    # ★ 공지: test 디렉토리 고정
SAMPLE_DIR  = Path("sample")
DEBUG       = os.getenv("DEBUG", "false").lower() in ("1", "true", "yes")

# LLM(옵션) – 기본 OFF: 재현성·결정성 유지
USE_LLM     = False
MODEL_PATH  = "models/Llama-3.1-8B-Instruct"
MAX_TOTAL_CHARS = 1200 # 출력 길이 상한 (가독/일관성)

# 가이드 준수: 향후 계획/외래 안내/일반 권고는 작성 제외
INCLUDE_FUTURE_PLAN = False # ★ 반드시 False (가이드 명시사항 준수)
# (선택) 과별 힌트 문구(부족 키워드 암시) – 공식 제출물은 편집성 문장 배제 권장
ADD_SPECIALTY_HINTS_OFFICIAL = False

# (선택) MeSH 2025 자산 1회 자동 빌드 트리거 (제출 안정성 위해 기본 False)
# – requested-data/ 경로에 MeSH XML이 존재할 때만 시도
BUILD_MESH_ASSETS_ONCE = False
```

→ LLM(옵션) – 기본 OFF: 재현성·결정성 유지

규칙/정규식 기반 파이프라인으로 데이터 경계 및 포맷 준수

평가 환경과 동일 동작 보장(랜덤성 X)

필요 시 모델 교체만으로 확장 가능(옵션)

→ USE_LLM=False, 길이 상한(MAX_TOTAL_CHARS), 향후계획 문구 미기재(INCLUDE_FUTURE_PLAN=False)로 가드레일 유지

2. ‘3-Stage’ Architecture

2.6 STAGE C – GUARDRAILS & EXPORT

```
def format_with_template_5(sections: dict) -> str:
    safe = _fill_required_5(sections)
    text = TEMPLATE_5.format(**safe).strip()
    text = re.sub(r"\n{3,}", "\n\n", text)
    return post_kor(text)
```

```
def run_submit_batch(pids: list[int], out_csv="result.csv"):
    """
    배치 제출 실행(본선 테스트셋 전체):
    - 전체 실행시간 측정해 runtime.txt에 저장 (속도 평가 대응)
    """
    rows = []
    t0 = _time.time()
    for pid in pids:
        txt = build_summary(int(pid))
        rows.append({"환자번호": int(pid), "summary": txt})
    pd.DataFrame(rows, columns=RESULT_COLS).to_csv(out_csv, index=False, encoding="utf-8-sig")
    Path("runtime.txt").write_text(f"({_time.time() - t0:.2f})", encoding="utf-8")

def write_runtime():
    """
    노트북 전체 실행시간(T0 기준) → runtime.txt 기록"""
    elapsed = _time.time() - T0
    Path("runtime.txt").write_text(f"{elapsed:.2f}", encoding="utf-8")
```

→ 5개 슬롯 이름·순서 1:1 고정 + 누락 시 ‘해당 항목 기재 없음’
자동 보정

RESULT.CSV와 RUNTIME.TXT를 일괄 생성
(배치 실행 시간 기록)

형식/사실 위반 차단, 제출 포맷 100% 일치

슬롯 존재·순서 보장 + 길이 상한으로 가독성 유지

2. ‘3-Stage’ Architecture

2.7 MESH 2025 통합

MeSH XML(desc/supp)



assets



힌트/라벨 정규화



추출

```
# MeSH 자산 자동 빌드(옵션) & 로드 -----
_ = _maybe_build_mesh_assets()
hints = _compile_hint_from_mesh()
IMG_HINT, FUNC_HINT, PATH_HINT = hints["IMG_HINT"], hints["FUNC_HINT"], hints["PATH_HINT"]
_MESH_LAB_ALIAS = _load_mesh_lab_alias()

# MeSH alias 우선 적용하도록 _canonize 재정의
def _canonize(name: str) -> str:
    nm = re.sub(r"\s+", " ", name).strip(" ")
    if nm.lower() in _MESH_LAB_ALIAS:
        return _MESH_LAB_ALIAS[nm.lower()]
    for pat, c in _SYNONYM_CANON:
        if pat.search(nm):
            return c
    return nm
```

→ **_MAYBE_BUILD_MESH_ASSETS() → ASSETS/MESH_***
생성 → _COMPILE_HINT_FROM_MESH()
_LOAD_MESH_LAB_ALIAS()로 런타임 반영

실패/미존재 시 기본 힌트로 동일 로직 수행

→ MESH XML이 있으면 힌트/라벨 자산을 자동 구성하여 탐지·라벨링 강화
자산이 없을 땐 기본 사전으로 자동 폴백(동일 코드 유지)

동의어·변형 확장으로 민감도 ↑, 오탐/혼용 ↓

데이터 가용성에 무관한 안정적인 동작

3. 리허설 정량 지표

3.1 형식·정확성



3. 리허설 정량 지표

3.2 임상적 유용성

입원시 진료과	환자번호_nunique	문서라인수_mean	문서라인수_median	문서라인수_max	총문자수_mean	총문자수_median	총문자수_max	총단어수(공백기준)_mean	총단어수(공백기준)_median	총단어수(공백기준)_max
순환기내과 (심장혈관센터)	100	35.040000	33.0	113	5842.700000	4736.5	19634	1017.050000	803.0	3738
신경외과 (뇌신경센터)	95	77.968421	77.0	162	9221.242105	8255.0	28313	1631.084211	1492.0	5040
신장내과	63	40.285714	36.0	111	8126.365079	5710.0	39551	1366.936508	947.0	6406
소화기센터	62	41.096774	37.5	97	6178.822581	5029.0	15732	1058.758065	887.5	2655
종합내과	33	44.787879	44.0	85	9926.515152	8724.0	21580	1655.303030	1494.0	3639
응급의학과	19	36.210526	33.0	77	6358.315789	5681.0	12941	1149.526316	1001.0	2417
외과	7	38.285714	41.0	49	5466.285714	4707.0	11346	938.142857	817.0	1822
신경외과 (척추센터)	6	78.833333	78.5	94	9764.000000	9494.5	13734	1757.000000	1690.5	2426
소화기내과 (암센터)	5	24.200000	22.0	47	4522.200000	2771.0	8130	780.400000	471.0	1344
신경과 (뇌신경센터)	4	43.500000	43.0	59	9414.750000	9120.5	12075	1705.750000	1592.5	2250
소화기내과	1	31.000000	31.0	31	3929.000000	3929.0	3929	677.000000	677.0	677
내분비대사내과	1	12.000000	12.0	12	723.000000	723.0	723	127.000000	127.0	127
심장혈관흉부외과 (폐센터)	1	101.000000	101.0	101	13905.000000	13905.0	13905	2566.000000	2566.0	2566
혈액종양내과 (암센터)	1	18.000000	18.0	18	5678.000000	5678.0	5678	957.000000	957.0	957
호흡기내과 (폐센터)	1	136.000000	136.0	136	32166.000000	32166.0	32166	5645.000000	5645.0	5645

과별 텍스트 통계 피벗 테이블

문서라인/총단어 중앙값 기준: 신경외과 > 순환기내과 순으로 분량이 큼
 변동성이 큰 과일수록 “핵심 키워드(예: EF/ECG/투석/내시경)”가 요약 품질을 좌우
 규칙 기반으로 중복 제거 + 최신값 ≤2개만 표기 → 장문/중복 리스크 최소화
 과 특성 반영 룰(심혈관/신장/신경/소화기)을 보강해 누락 리스크 ↓

▶ 분량은 과별 편차가 커서 평균보다는 중앙값을 기준으로
 → 각 과의 대표 키워드를 룰에 반영했고, 수치·검사는 최신
 1~2개만 표기해 장문과 중복을 잡음

4. 리스크 & 대응

운영 안전장치 사전 반영

리스크 항목	대응 방식	코드/설명
MeSH 자산 미탑재	기본 사전 폴백	_load_mesh_lab_alias() → 자산 없을 시 internal dictionary 사용
섹션 결측	자동 보정 문구	_fill_required_5() → "해당 항목 기재 없음" 자동 삽입
장문/중복	길이 상한·최신값 ≤2	MAX_TOTAL_CHARS, dedup_keep_order()
과별 핵심 힌트 누락	enforce_specialty_points	과별 핵심항목 자동 주입(공식 제출 OFF)
환경 차이/인코딩	read_csv_safe()	UTF-8/UTF-8-SIG 자동 판별, 예외 메시지 반환



오류·복구 시나리오 디벨롭

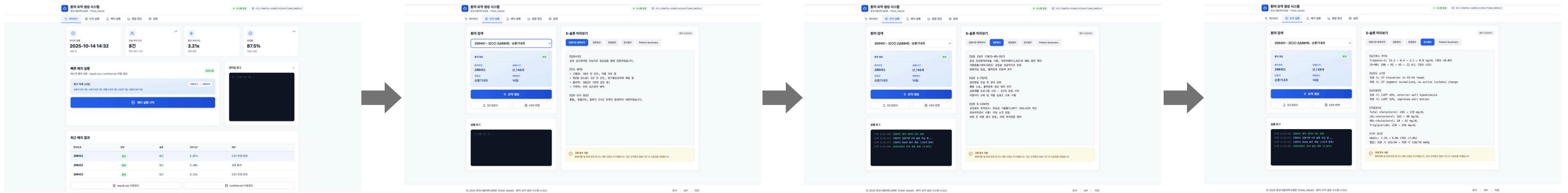
- 입력 결측/포맷 오류: `read_csv_safe/스키마 밸리데이션` → 400 반환 + 필드별 원인
- 날짜 오류: `get_stay()` 유효성 실패 시 명시적 에러(입·퇴원 일 결측/역전)
- 섹션 데이터 부족: 슬롯 자동 보정(형식 유지) → EMR에 경고 배지만 노출
- 외부 저장 실패(대회 전용 S3): 자동 재시도 → 실패 시 로컬 보존 + 운영자 알림
- 성능 이슈: 타임아웃(예: 10s) 초과 시 캐시된 마지막 성공본 제공 옵션

→ 현실적 장애 포인트를 사전에 제거해서, 평가 기준(형식·일관성·정확성·효율성)에 안전하게 맞췄습니다

평가에 영향을 줄 만한 리스크는 전부 가드레일로 막았습니다
특히 퇴원서식 혼입은 실행 단계에서 자동 제거합니다

Appendix

분당서울대병원 시스템 도입 구체안



메인보드

입원사유/병력요약

입원결과

검사결과

배포 아키텍처

형태: 내부망 Dockerized microservice(Python 3.10 + pandas/numpy + 경량 regex 엔진)

스케일링: 1-2 pod로 시작, HPA(QPS/CPU 기반)

스토리지: 임시 처리(메모리/로컬) → 로그·결과는 병원 로컬 볼륨만 사용(외부 전송 없음)

EMR 연계: API Gateway 혹은 내부 Nginx Ingress로 라우팅

엔진 특성

결정성: 랜덤 요소 없음(LLM OFF), 동일 입력 → 동일 출력

라벨 표준화: 수치/검사 동의어→정규 라벨(예: Cr→Creatinine (Cr), Hb→Hemoglobin (Hb), Na/K/혈소판 등) → 중복 억제

기간 일관성: `within_stay()`로 테이블별 날짜 컬럼을 변환·검증 후 입원 구간 밖 데이터 배제

MeSH 2025 연동: 자산 존재 시 탐지/라벨링 강화, 없으면 기본 사전으로 폴백(런타임 플래그 無)

Appendix

분당서울대병원 시스템 도입 구체안

도입 배경

- 퇴원기록지 자동생성 엔진을 실제 EHR 환경에 적용할 수 있도록, 분당서울대병원(EMR-PACS 통합 구조)에서의 시스템 연동 프로토타입 설계
- 목표: 비임상 테스트 → 파일럿 운영으로 확장 가능한 구조 마련



연동 방식 개요

- 입·퇴원·진단·수술·간호·검사 테이블 → API Gateway → 텍스트 엔진 호출 → result.csv 반환
- submit.ipynb 내 핵심 함수(run_submit_batch)를 백엔드 API로 래핑
- 병원 내부망 내 Dockerized microservice 형태로 배포 계획
- 보안: 모든 로깅/데이터 처리 로그는 병원 내 로컬 볼륨으로만 저장

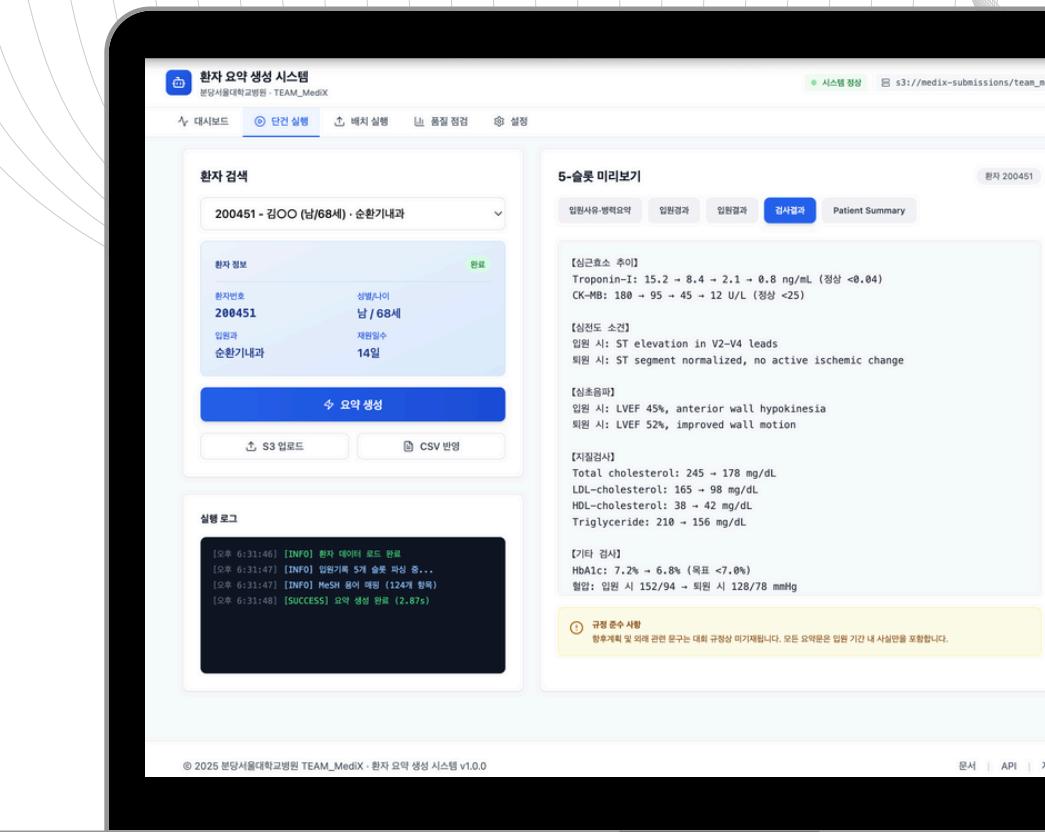
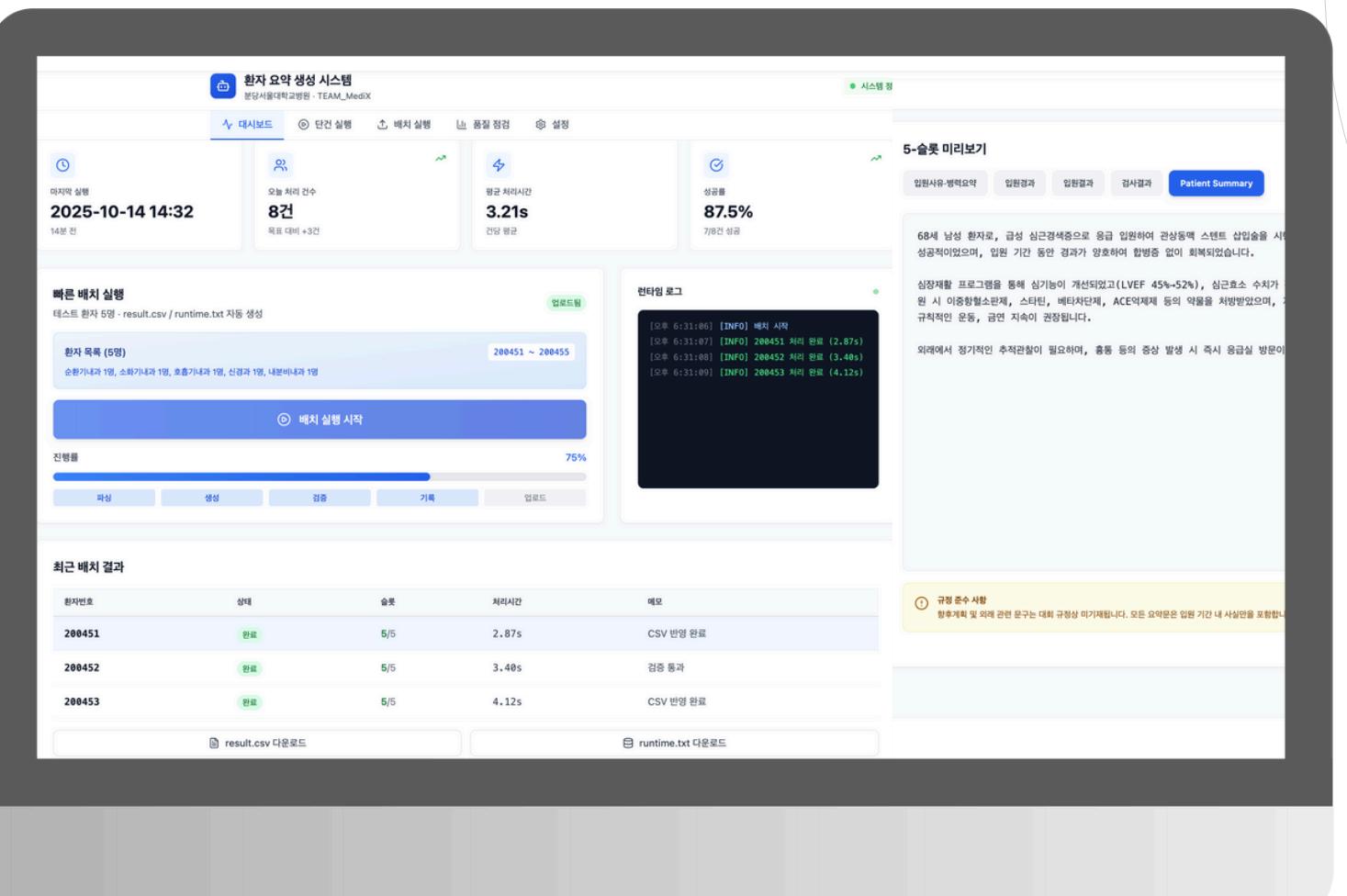
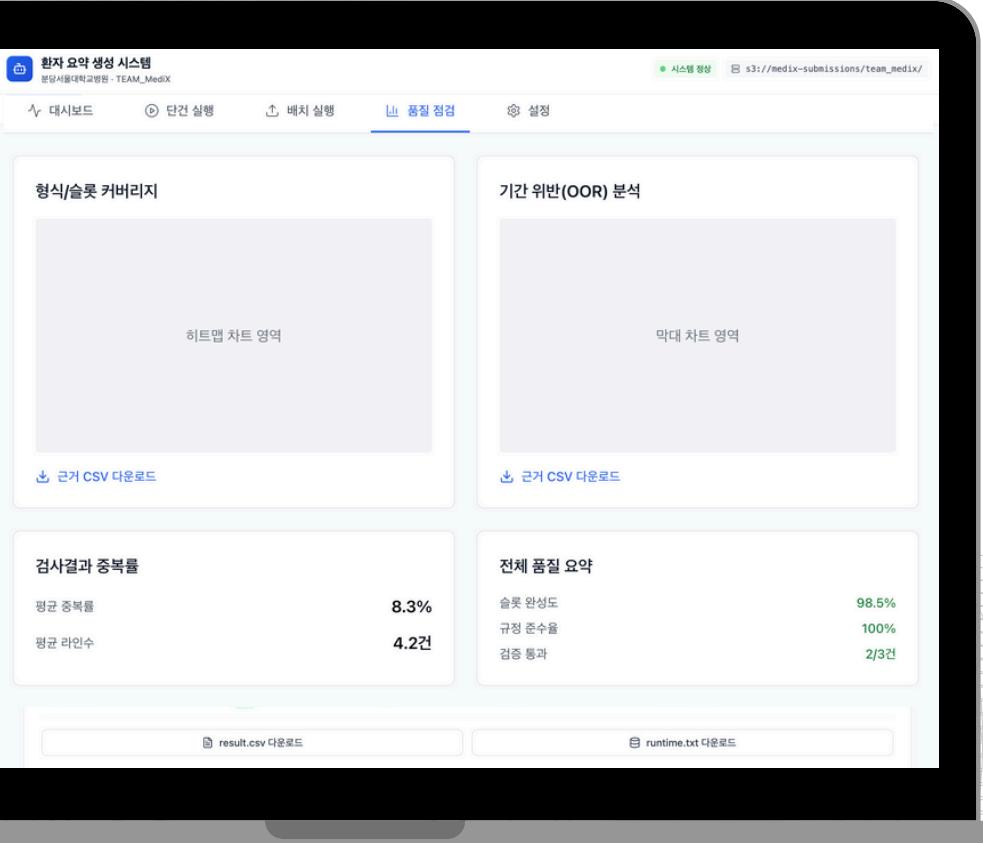
데이터 연동 흐름 (EMR 내부)

- EMR에서 환자번호, 입·퇴원일 조회 → 원장 테이블 조인(진단/수술/간호/의무기록/초진)
- 마이크로서비스 /summarize 호출(POST, JSON)
- 엔진이 Stage A(추출) → Stage B(옵션·기본 OFF) → Stage C(가드레일/Export) 실행
- 응답: result.summary_text(5-슬롯), result.runtime_sec, result.hash_id
- EMR ‘퇴원요약’ 초안 필드에 바로 주입 + 감사 로그 저장

M.A.R.S. 2025 SNUBH 의무기록 생성 데이터톤

Medical Auto-documentation with Real-world Structuring:
LLM Clinical Note Generation Challenge

입원~퇴원 구간의 임상기록에서 규칙·정규식 + MeSH 정규화로 핵심 정보를 뽑아
LLM 없이도 결정적으로 필수 5슬롯 퇴원요약을 자동 생성하는 파이프라인



M.A.R.S. 2025 SNUBH 의무기록 생성 데이터톤

Medical Auto-documentation with Real-world Structuring:
LLM Clinical Note Generation Challenge

본선 결과 보고서 [MediX]

- 감사합니다 -