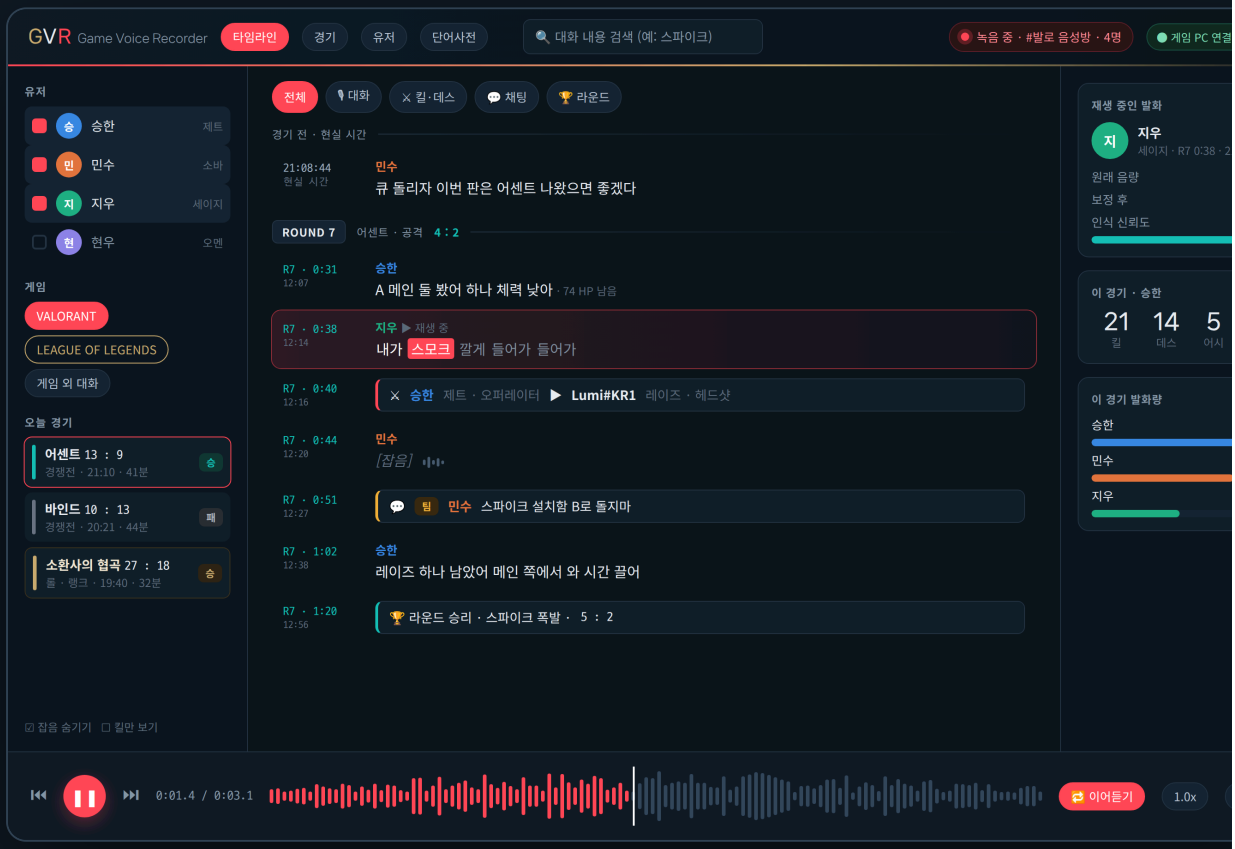
Game Voice Recorder

게임 연동 디스코드 음성 기록 봇 + 기록 열람 사이트
프로젝트 보고서 (질문 답변 반영 설계안)

문서 버전	v0.3 (확인 질문 Q1~Q11 답변 반영, 화면 시안 갱신)
작성일	2026-10-14
저장소	git.tkrmagid.kr/tkrmagid/game_voice_recorder
1차 대상 게임	VALORANT · League of Legends
실행 형태	Windows 포터블 exe (게임 PC) + 사이트 서버
게임 PC GPU	NVIDIA GeForce RTX 5070 Ti (16GB)

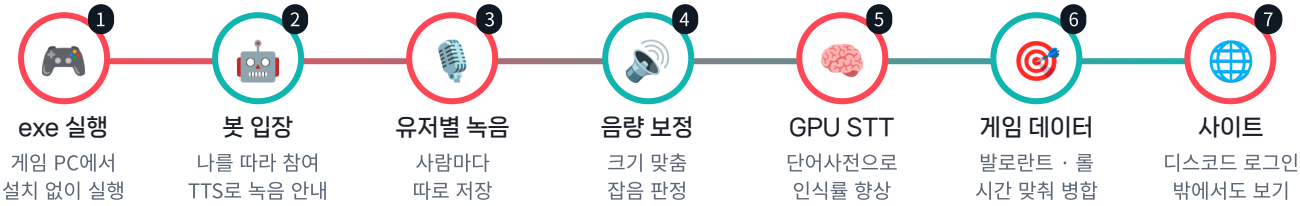


v0.2의 확인 질문에 대한 답변을 설계에 반영한 문서입니다. 수치(VRAM, 용량 등)는 추정치이며 기술 검증 단계(M0)에서 실측 후 확정합니다.

01 요약

디스코드 음성채널에 봇이 참여해 참가자 각자의 음성을 따로 녹음하고, 게임 PC의 GPU(RTX 5070 Ti)로 음성을 한국어 텍스트로 바꾼 뒤(STT), 발로란트·리그 오브 레전드 경기 데이터(킬·데스·라운드·오브젝트·인게임 채팅 등)와 같은 타임라인에 묶어 웹사이트에서 보고 들을 수 있게 하는 프로그램입니다. 게임 PC에서는 설치 없이 실행하는 포터블 exe 하나로 봇, STT 엔진, 기록 저장소, 설정 화면이 돌아갑니다. 친구들은 별도 사이트 서버에 디스코드 계정으로 로그인해 밖에서도 기록을 봅니다. 기록은 게임 PC에만 있고, 사이트 서버는 게임 PC 프로그램과 연결된 동안에만 화면을 보여 줍니다.

한눈에 보는 흐름



100%
화자 구분 정확도
디스코드가 유저별로 음성을 따로 보냄

10명
경기 데이터 범위
같은 경기 10명 전원의 킬·데스

≈14MB
말한 시간 1시간당 음성 용량
(Opus 32kbps 기준)

7일
보관 기간
지나면 자동 삭제

핵심 기능

🎙️ 음성 기록

디스코드 봇이 음성채널에 들어가 유저별로 음성을 따로 받아 발화 단위로 저장합니다. 화자는 디스코드가 유저별로 스트림을 나눠 보내 주기 때문에 추정하지 않고 정확히 구분됩니다.

📝 GPU STT (한국어 고정)

faster-whisper(Whisper 계열) 모델을 NVIDIA GPU로 구동합니다. RTX 5070 Ti(16GB) 기준으로 경기 중에는 large-v3-turbo로 바로 변환하고, 경기가 끝나면 large-v3로 다시 변환해 정확도를 높입니다.

🔊 음량 보정 · 잡음 처리

작은 목소리와 큰 목소리를 같은 크기로 맞춘 뒤 인식하고, 말이 아닌 소리는 [잡음]으로 표시합니다.

🎯 발로란트 · 롤 연동

발로란트는 라운드, 킬·데스·어시스트(시각·위치·무기), 점수, 인게임 채팅을, 롤은 킬·오브젝트(드래곤·바론·포탑)·골드·CS를 음성 기록과 같은 시간축에 배치합니다.

🕒 이중 시간 표시

발로란트 경기 중 발화는 라운드 시간과 경기 경과 시간을 함께(예: `R7 · 0:42 + 12:18`), 롤은 게임 시간(`21:14`)으로, 경기 밖 발화는 현실 시간으로 표시합니다.

🌐 기록 열람 사이트

디스코드로 로그인한 서버 멤버만 들어옵니다. 문장을 누르면 그 위치부터 재생되고 재생 중인 단어가 강조됩니다. 게임 PC 프로그램과 연결이 끊기면 화면을 가리고 "연결 해제됨"을 표시합니다.

답변으로 확정된 내용

질문	답변	설계에 반영한 내용	장
Q1	B (로컬 API)	발로란트·롤 모두 게임 PC의 Riot 클라이언트 로컬 API로 구현합니다. 공식 API(A)가 꼭 필요한 기능이 생기면 그때 키 발급을 따로 요청드립니다.	6
Q2	C (외부 접속) + 연결 확인	사이트는 별도 사이트 서버에서 운영합니다. 서버가 게임 PC 프로그램과 계속 연결을 확인하고, 등록된 프로그램이 연결되어 있지 않으면 사이트를 가린 뒤 "프로그램 연결이 해제됨"을 표시합니다.	3 · 7
Q3	NVIDIA만 (RTX 5070 Ti)	faster-whisper(CUDA) 하나만 씁니다. AMD·Intel용 엔진은 만들지 않습니다. 16GB VRAM 기준으로 모델을 정했습니다.	5
Q4	a	경기 중 실시간 변환 후, 경기가 끝나면 큰 모델로 다시 변환해 결과를 바꿉니다.	4 · 5
Q5	c (둘 다)	발로란트는 `R7 · 0:42`와 경기 경과 시간 `12:18`을 함께 표시합니다. 롤은 라운드가 없어 게임 시간만 표시합니다.	6
Q6	a + b	핫워드·교정 사전에 더해 실제 모델 파인튜닝(LoRA)까지 개발 범위에 넣었습니다.	5 · 10
Q7	b + c	내가 들어간 음성채널을 따라가고, `/녹음시작` · `/녹음종료` 명령으로도 부를 수 있습니다.	8
Q8	TTS 음성 안내	채팅 메시지는 보내지 않습니다. 봇이 음성채널에 들어가면 TTS로 녹음 사실을 말하고, `/녹음제외`로 빠지고 `/녹음참여`로 다시 참여합니다.	7 · 8
Q9	7일 후 자동 삭제	기록은 7일이 지나면 자동으로 지웁니다. 최대 저장 용량은 약 0.4GB입니다.	9
Q10	c (디스코드 로그인)	봇이 있는 디스코드 서버의 멤버만 사이트에 들어올 수 있습니다.	3 · 7
Q11	한국어 고정 · 발로+롤 디자인 · 롤 추가	인식 언어는 한국어로 고정하고, 디자인은 발로란트(레드·HUD)와 롤(골드·헥스테크) 분위기를 섞습니다. 리그 오브 레전드 연동을 1차 버전에 함께 넣습니다.	6 · 7

> **구현 전에 추가로 확인할 것 (12장 A1~A5)**
> 사이트 서버를 둘 곳과 주소, 7일 삭제 범위, 파인튜닝용 학습 데이터를 7일 넘게 보관할지, 봇 토큰과 사용할 디스코드 서버, 롤 인게임 채팅 제외 여부를 확인해야 합니다.

목차

1. 요약
2. 요구사항 정리
3. 시스템 구성
4. 음성 처리 파이프라인 (음량 보정 · 잡음 · STT)
5. STT 모델 · GPU · 단어사전 · 파인튜닝
6. 게임 연동 (발로란트 · 리그 오브 레전드)
7. 사이트 기능 · 접속 · UI/UX 콘셉트
8. exe 설정 페이지
9. 데이터 저장 구조 · 용량
10. 기술 스택 · 개발 일정
11. 위험 요소 · 대응
12. 질문 답변 정리 · 추가 확인 사항

02 요구사항 정리

요청 내용과 Q1~Q11 답변을 기능 단위로 묶고 번호를 붙였습니다. 우선순위: 필수 1차 버전에 반드시 포함 · 권장 1차에 포함하되 범위 조정 가능 · 확장 2차 이후

ID	영역	요구사항	우선
F-01	디스코드 봇	설정한 봇 토큰으로 접속해, 내가 들어간 음성채널을 따라가거나 /녹음시작 명령으로 참여하고, 참여자별 음성을 따로 수신	필수
F-02	음성 저장	발화 단위로 실제 음성 파일을 저장하고 사이트에서 재생	필수
F-03	STT	NVIDIA GPU 기반 한국어 음성→텍스트 변환, 단어별 시각(word timestamp) 포함	필수
F-04	음량 보정	유저별 음량 차이를 보정해 같은 크기로 맞춘 뒤 STT	필수
F-05	잡음 표시	말이 아닌 소리(키보드, 숨소리, 게임 소리 등)는 [잡음]으로 표시	필수
F-06	단어사전	게임·일상·대학생·20대·발로란트·롤 단어사전(핫워드 + 교정 사전)과 모델 파인튜닝으로 인식률 향상	필수
F-07	시간 표시	경기 중에는 인게임 시간(발로란트: 라운드 시간 + 경기 경과 시간, 롤: 게임 시간), 경기 밖에서는 현실 시간	필수
F-08	기록 목록	시간 · 화자 · 텍스트 표시, 클릭하면 해당 음성 재생	필수
F-09	따라 읽기 재생	재생 위치에 맞춰 단어 단위 강조, 단어 클릭 시 그 지점으로 이동, 다음 발화 자동 이어듣기	필수
F-10	필터	유저별, 게임별(+경기별, 날짜, 키워드 검색) 필터	필수
F-11	발로란트 연동	경기 시간, 킬·데스·어시, 라운드 결과, 점수, 인게임 채팅 등을 STT와 함께 표시 (Riot 클라이언트 로컬 API)	필수
F-12	모델 관리	모델 다운로드 위치·크기 선택, GPU와 게임 사양에 맞춘 자동 추천·다운로드	필수
F-13	설정 페이지	exe 안에서 봇 토큰 등 각종 설정 입력·저장	필수
F-14	UI/UX	발로란트 + 리그 오브 레전드 분위기의 디자인, 애니메이션, 효과음 등 부가 효과	권장
F-15	롤 연동	리그 오브 레전드의 게임 시간, 킬·데스·어시, 오브젝트, 골드·CS를 STT와 함께 표시. 게임 연동부는 어댑터 구조로 나눠 이후 다른 게임도 추가 가능	필수
F-16	사전 개선 루프	사이트에서 잘못 인식된 텍스트를 고치면 교정 사전과 파인튜닝 학습 데이터에 자동 반영	필수
F-17	외부 접속 · 연결 확인	사이트 서버에서 사이트를 운영하고, 게임 PC 프로그램과 연결을 계속 확인. 등록된 프로그램이 연결되어 있지 않으면 사이트를 가리고 "프로그램 연결이 해제됨" 표시	필수
F-18	디스코드 로그인	봇이 있는 디스코드 서버의 멤버만 사이트 접속	필수
F-19	녹음 안내 · 제외	봇이 음성채널에 들어가면 TTS 음성으로 녹음 사실 안내(채팅 메시지 없음). /녹음제외 · /녹음참여로 본인이 직접 빠지거나 다시 참여	필수
F-20	파인튜닝	교정 데이터와 합성 음성으로 STT 모델을 LoRA 방식으로 추가 학습하고, 학습한 모델로 교체	권장

비기능 요구사항

ID	항목	내용
N-01	배포 형태	Windows 10/11 x64 포터블 실행 파일. 설치 과정이나 관리자 권한 없이 실행
N-02	게임 성능 · GPU	NVIDIA GPU 전용(기준: RTX 5070 Ti 16GB). STT가 게임 프레임을 떨어뜨리지 않도록 VRAM 예산 제한, 낮은 프로세스 우선순위, "경기 중 실시간 / 경기 후 정밀 재변환" 구성
N-03	보안	봇 토큰 등 비밀값은 Windows DPAPI로 암호화해 저장. 사이트는 HTTPS + 디스코드 로그인 필수. 게임 PC는 등록 코드로 한 번 등록한 뒤 기기 키로만 사이트 서버에 연결
N-04	개인정보	녹음 사실 TTS 음성 안내(봇 입장 시, 새 사람이 들어올 때), /녹음제외 유저의 음성은 저장하지 않음, 7일 지나면 자동 삭제
N-05	안정성	STT 처리가 밀려도 녹음은 계속되도록 녹음과 STT를 분리(대기열), 프로그램 재시작 시 미처리분 이어서 처리

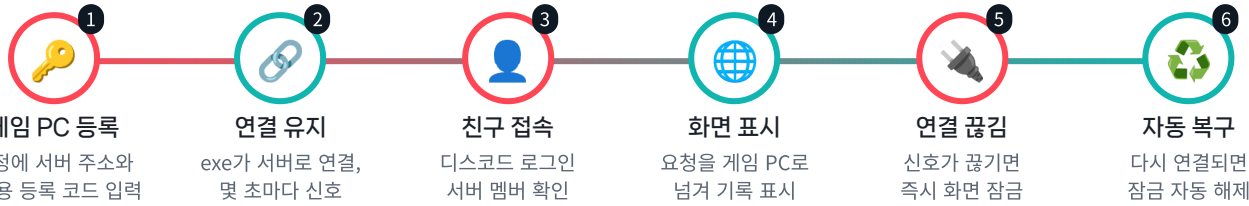
03 시스템 구성

게임 PC에서 실행되는 포터블 앱과, 친구들이 접속하는 사이트 서버 두 부분으로 나뉩니다. 녹음·게임 연동·기록 저장은 게임 PC의 Node.js 코어가 맡고, GPU STT는 별도 Python 워커 프로세스가 맡습니다. 사이트 서버는 로그인과 화면 제공, 게임 PC로 요청을 넘기는 중계만 하고 기록은 저장하지 않습니다.



Discord	Riot 클라이언트 (게임 PC)	Riot 공식 API (필요할 때만)	Hugging Face
게이트웨이 · 음성 서버 (E2EE) · OAuth2 로그인	발로란트 로컬 API · 롤 게임 클라이언트 API(2999 포트) · 롤 클라이언트(LCU) API	필요한 기능이 생기면 키 발급을 따로 요청	STT 모델 다운로드

연결 확인 · 사이트 잠금 흐름



"프로그램 등록"은 설정 화면에서 게임 PC를 사이트 서버에 한 번 등록하는 것으로 해석했습니다. 등록된 기기가 없거나, 등록된 기기가 연결되어 있지 않으면 같은 잠금 화면(7-4)을 보여 줍니다.

왜 이렇게 나눴나

- **디스코드 음성 수신은 Node.js**: 2026년 3월부터 디스코드 음성은 종단간 암호화(DAVE)를 지원하지 않는 클라이언트·봇의 참여를 막습니다. 수신(receive)과 DAVE를 함께 지원하는 라이브러리 중 가장 검증된 것이 @discordjs/voice입니다.
- **STT는 Python 워커**: GPU STT 생태계(faster-whisper, CTranslate2, Silero VAD)가 Python 쪽이 가장 성숙합니다. 별도 프로세스라 GPU 메모리를 따로 관리하고 문제가 생기면 워커만 재시작할 수 있습니다.
- **CUDA 런타임과 모델은 첫 실행 때 내려받기**: CUDA 라이브러리(cuBLAS·cuDNN)만 수백 MB~1GB 이상이라 exe에 넣지 않습니다. exe 크기를 줄이고, RTX 5070 Ti(Blackwell)에는 CUDA 12.8 이상 런타임이 필요합니다.
- **사이트 서버는 중계만, 기록은 게임 PC에**: 게임 PC가 서버 쪽으로 먼저 연결하기 때문에 공유기 포트를 열 필요가 없습니다. 기록을 서버에 올리지 않으므로 게임 PC가 꺼지면 사이트도 볼 수 없는데, 이 상태를 잠금 화면으로 분명하게 보여 줍니다(Q2 답변).
- **로그인은 디스코드로**: 녹음 대상이 디스코드 서버 멤버이므로, 같은 서버 멤버만 기록을 보게 하는 것이 가장 자연스럽습니다(Q10 답변).

> **포터블 exe 참고**
> Electron의 단일 파일 포터블 exe는 실행할 때마다 임시 폴더에 압축을 풀기 때문에 첫 화면이 몇 초 늦게 뜰 수 있습니다. 대안으로 "폴더째 들고 다니는 포터블(zip 압축 해제 후 exe 실행)" 형태도 가능합니다. 이 경우 실행이 빠르고 모델·기록을 같은 폴더에 둘 수 있습니다.

04 음성 처리 파이프라인

1 수신	2 발화 분할	3 원본 저장	4 음량 정규화	5 VAD	6 STT	7 후처리	8 병합
유저별 Opus 스트림 (48kHz)	무음 0.8초에서 끊음, 최대 30초	Opus(.ogg) 그대로	발화별 라우드니스 맞춤 + 리미터	Silero VAD로 말소리 구간 판정	faster-whisper + 핫워드	교정 사전 · 환각 제거 · [잡음]	게임 이벤트와 타임라인 정렬

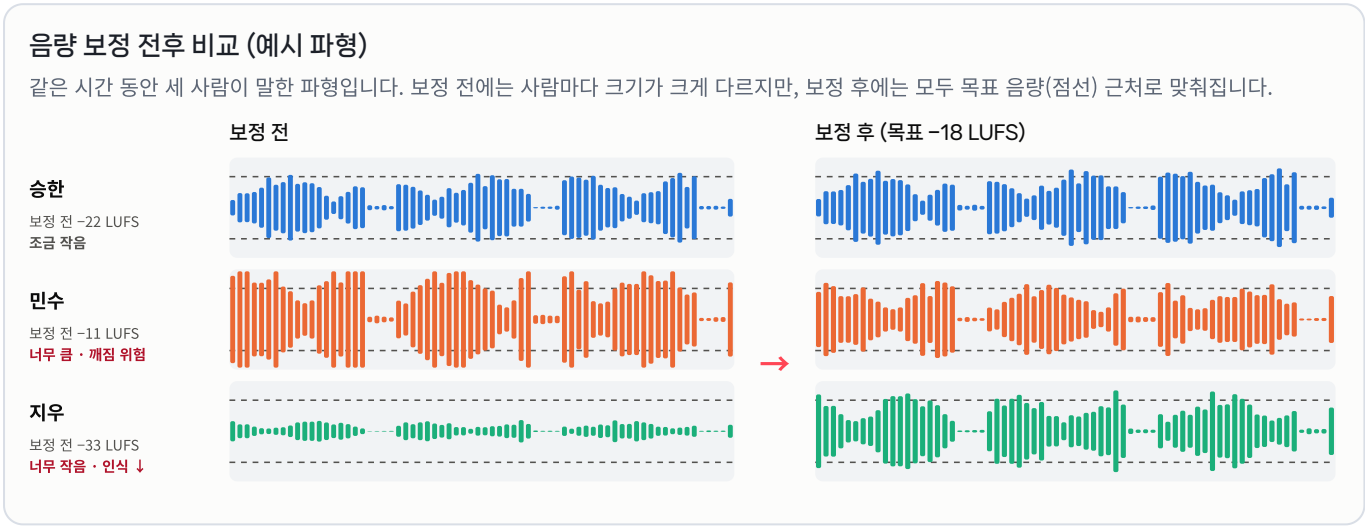
4-1. 유저별 수신과 화자 구분

디스코드는 음성채널 참가자마다 별도 스트림을 보내므로 화자 분리(diarization) 모델이 필요 없고, 누가 말했는지 100% 정확합니다. 여러 명이 동시에 말해도 서로 섞이지 않습니다. 발화 시작 시각은 PC 시계 기준 밀리초 단위로 기록합니다.

4-2. 음량 보정 (작은 목소리 · 큰 목소리)

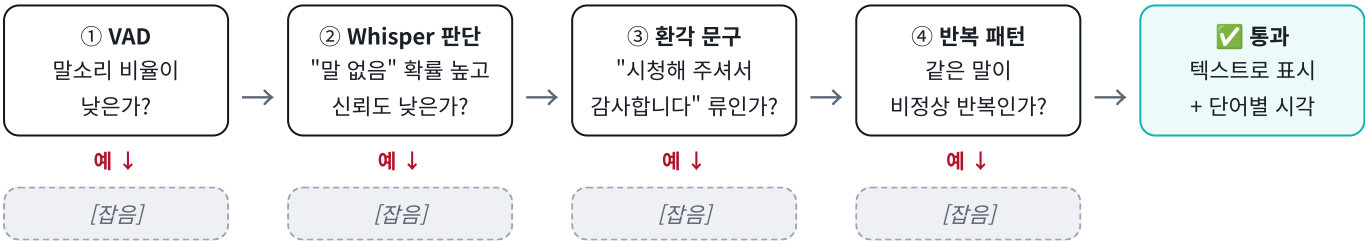
- **발화 단위 라우드니스 정규화**: 각 발화의 체감 음량(LUFS)을 측정해 목표값(약 −18 LUFS)으로 끌어올리거나 낮춥니다. 단순 최대값(peak) 기준이 아니라 사람이 듣는 크기 기준이라 작은 목소리도 충분히 커집니다.

- **유저별 게인 프로필**: 유저마다 평균 음량을 누적 학습해, 매우 짧은 발화("어", "ㅇㅇ")처럼 측정이 불안정한 경우에도 그 유저의 평소 보정값을 씁니다.
- **리미터 · 클리핑 방지**: 크게 키울 때 소리가 깨지지 않도록 마지막에 리미터를 겁니다.
- **제외 유저**: /녹음제외를 쓴 사람의 음성은 받는 즉시 버리고 파일로 남기지 않습니다.
- **선택: 잡음 억제**: 마이크 잡음이 심한 유저를 위해 노이즈 억제(RNNoise 계열)를 STT 입력에만 선택적으로 적용합니다.
- **저장 정책**: 원본은 그대로 보관하고, 사이트 재생 시에는 정규화한 음량으로 들리게 합니다(재생 시 게인 적용 → 파일을 두 벌 저장할 필요 없음).



4-3. 잡음 판정 → [잡음]

발화 하나가 들어오면 아래 순서대로 검사합니다. 어느 단계에서든 "예"가 나오면 [잡음]으로 표시하고, 네 단계를 모두 통과해야 텍스트로 보여 줍니다.



②는 모델이 계산한 무음 확률(no_speech_prob)과 평균 로그 확률(avg_logprob)을 씁니다. ③은 한국어 Whisper가 키보드 소리·숨소리 같은 잡음에서 자주 지어내는 문장 목록입니다. 잡음으로 판정된 발화도 음성 파일은 남기고 목록에는 흐리게 표시하며, 사이트에서 "잡음 숨기기"로 감출 수 있습니다. 판정 기준값은 실제 녹음으로 조정합니다.

4-4. 실시간 처리와 게임 성능

녹음은 항상 실시간으로 하고, STT는 대기열로 처리합니다. Q4 답변(a)에 따라 **"경기 중 실시간 변환(large-v3-turbo) → 경기 종료 후 같은 구간을 large-v3로 다시 변환해 결과 교체"**로 확정했습니다. 사이트에서는 경기 중에도 몇 초 안에 텍스트가 올라오고, 경기가 끝난 뒤 문장이 더 정확한 결과로 바뀝니다. 인식 언어는 한국어로 고정해 언어 감지 단계를 건너뜁니다(감지 오류가 없고 조금 더 빠릅니다).

05 STT 모델 · GPU · 단어사전 · 파인튜닝

5-1. 엔진

faster-whisper(CTranslate2 기반 Whisper 구현)를 기본 엔진으로 씁니다. 한국어 인식이 좋고, GPU에서 원본 Whisper보다 빠르고 메모리를 적게 쓰며, 단어별 시각과 핫워드(hotwords) 기능을 지원합니다. Q3 답변에 따라 NVIDIA GPU만 지원하고, 인식 언어는 한국어(`language="ko"`)로 고정합니다.

5-2. 모델 구성 (RTX 5070 Ti 16GB 기준)

NVML로 GPU 이름과 VRAM을 읽고, 게임이 쓰는 VRAM을 먼저 뺀 **"STT 예산"** 안에서 모델을 고릅니다. 지금 게임 PC는 16GB라 여유가 충분해서, 경기 중 모델과 경기 후 정밀 모델을 따로 둡니다.

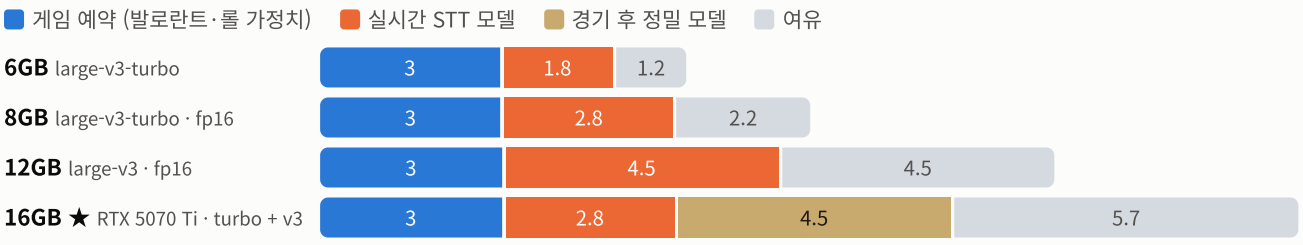
용도	모델	연산 정밀도	VRAM (추정)	비고
경기 중 실시간	large-v3-turbo	float16	약 2.8GB	속도 우선. 게임 프레임 영향을 줄이려고 낮은 우선순위로 실행
경기 후 정밀	large-v3	float16	약 4.5GB	경기가 끝난 뒤에만 올려 같은 구간을 다시 변환
파인튜닝	large-v3-turbo + LoRA	bf16	약 10~14GB	게임하지 않을 때만 실행 (5-5)

> **RTX 50 시리즈(Blackwell) 주의점**
> RTX 5070 Ti는 CUDA 12.8 이상과 CTranslate2 4.5 이상이 필요합니다. 50 시리즈에서는 int8 계열 연산 모드가 cuBLAS 오류를 낸 사례가 있어 기본값을 float16으로 둡니다. 기술 검증(M0)에서 실제로 돌려 속도와 VRAM을 확인하고, 문제가 있으면 CTranslate2를 이 GPU용으로 직접 빌드합니다.

게임별 VRAM 예약값은 게임 프로필로 관리합니다. 발로란트·롤 모두 비교적 가벼운 게임이라 둘 중 큰 값인 약 3GB를 우선 가정하고, 실측 후 조정합니다. 다른 NVIDIA GPU를 쓰는 경우를 위해 아래처럼 VRAM 크기별 규칙도 남겨 둡니다. 수치는 모두 추정치입니다.

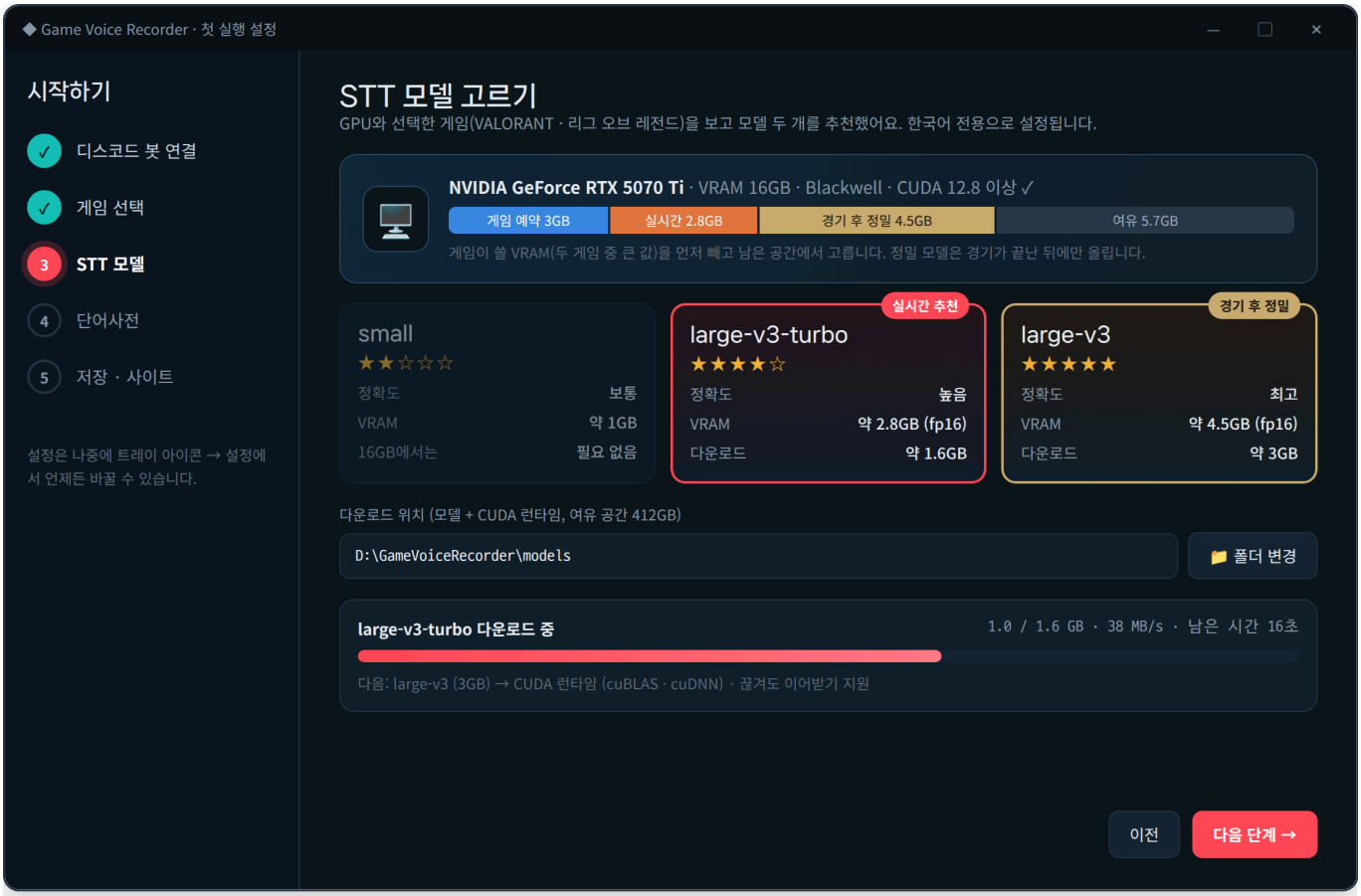
GPU VRAM 별 배분 (추정, 단위 GB)

막대 길이 = GPU 전체 VRAM. 게임 몫을 먼저 떼어 두고, 남은 공간에 들어가는 가장 좋은 모델을 고릅니다. 16GB(현재 게임 PC)는 실시간 모델과 정밀 모델을 함께 둡니다.



5-3. 모델 다운로드

- 첫 실행 마법사에서 **다운로드 폴더 선택 → GPU 감지 결과·추천 모델 표시 → 크기 선택 → 다운로드** 순서로 진행합니다.
- 이어받기, 해시 검사, 디스크 여유 공간 확인, 진행률·남은 시간 표시를 지원합니다.
- CUDA 런타임(cuBLAS·cuDNN)도 같은 폴더에 함께 받습니다. 그래픽 드라이버 버전이 낮으면 업데이트 안내를 띄우고, NVIDIA GPU가 없으면 지원하지 않는다고 안내합니다.



화면 예시 · 첫 실행 마법사 3단계 (STT 모델 고르기) RTX 5070 Ti(16GB)를 감지해 실시간용 large-v3-turbo와 경기 후 정밀용 large-v3를 함께 추천하고, VRAM 배분과 다운로드 진행률을 보여 줍니다. 시안

5-4. 단어사전으로 인식률 높이기

Q6 답변(a + b)에 따라 세 단계를 모두 넣습니다. 1·2단계는 처음부터 적용하고, 3단계(파인튜닝)는 실사용 데이터가 어느 정도 쌓인 뒤 시작합니다.

단계	방식	비용 · 효과
1. 핫워드 주입	변환할 때마다 상황에 맞는 단어 목록을 모델에 힌트로 줍니다. 경기 중이면 발로란트 요원·맵·스킬·콜아웃, 경기 밖이면 일상·대학생 단어를 우선합니다.	즉시 적용 · 고유명사 인식 개선
2. 교정 사전	자주 틀리는 표현을 바로잡는 규칙입니다(예: "레이저" → "레이즈", "스파이크 설치" 표기 통일). 한글 자모 유사도로 사전 단어와 가까운 오인식을 찾아 고치고, 사이트에서 사용자가 고친 내용도 사전에 쌓입니다.	즉시 적용 · 사용할수록 개선
3. 파인튜닝	단어사전 문장을 음성 합성(TTS)으로 만들고, 실제 녹음 중 사용자가 교정한 문장을 모아 LoRA 방식으로 모델을 추가 학습합니다. 자세한 내용은 5-5에 있습니다.	1~2주 작업 · 데이터가 쌓일수록 개선

단어사전 적용 예시

같은 발화를 사전 없이 인식한 결과와 핫워드 + 교정 사전을 적용한 결과 비교 (예시)

사전 없음	"레이저 궁 빠졌어 비 싸이트 로태 하자 나 오퍼래이터 삼"
사전 적용	"**레이즈** 궁 빠졌어 **B 사이트** **로테** 하자 나 **오퍼레이터** 삼"

빨간 밑줄: 사전 없이 틀리기 쉬운 부분 · 초록 바탕: 핫워드와 교정 사전으로 바로잡힌 부분

기본 제공 사전 (예시)

발로란트

요원 제트, 레이즈, 세이지, 소바, 오멘, 킬조이, 사이퍼, 스카이, 요루, 네온, 페이드, 하버, 게코, 클로브 …
맵·콜아웃 어센트, 바인드, 헤이븐, 스플릿, 로터스, 선셋 … A사이트, 미드, 헤븐, 로비, 메인, 롱, 숏, 링크
용어 스파이크, 설치, 해체(디퓨즈), 에코, 풀바이, 세이브, 오퍼(오퍼레이터), 밴달, 팬텀, 셰리프, 원탭, 에이스, 클러치, 로테

게임 공통 · 일상 · 20대/대학생

게임 공통 킬, 데스, 어시, 힐, 쿨, 궁, 빽, 라인, 랭겜, 티어, 큐 돌리자, 트롤, 던짐, 캐리
일상·20대 레전드, 미쳤다, 개웃기네, 인정, 에바, 킹받네, 갓생, 존맛, 퇴근각
대학생 과제, 팀플, 중간고사, 기말, 휴강, 학식, 에타, 종강, 개강, 학점, 동아리, 엠티

기본 사전은 수천 단어 규모로 준비하고, 사이트·설정에서 직접 추가·수정할 수 있게 합니다. 실제 목록은 구현 단계에서 공식 게임 데이터(발로란트 요원·맵·무기, 롤 챔피언·아이템·스킬 이름)를 기반으로 자동 생성·갱신합니다. 롤 사전 예시: 아리, 리 신, 쓰레쉬, 이즈리얼 … 바론, 드래곤, 전령, 유충, 억제기, 갱, 백도어, 텔, 점멸, 강타, 와드, 딜교, 라인전, 한타.

5-5. 파인튜닝 계획

항목	내용
학습 데이터	① 사이트에서 사용자가 고친 문장과 그 음성 조각 ② 단어사전 문장을 한국어 TTS로 합성한 음성. /녹음제외 유저의 음성은 쓰지 않습니다.
방식	large-v3-turbo에 LoRA를 붙여 학습하고, 결과를 CTranslate2 형식으로 바꿔 실시간 모델로 씁니다.
학습 장소	게임 PC(RTX 5070 Ti 16GB). 게임이 실행 중이면 시작하지 않고, 게임이 켜지면 바로 멈춥니다. 설정에서 "지금 학습" 또는 "매일 새벽 자동"을 고릅니다.
적용 기준	학습 전후로 같은 검증 문장의 오류율(CER)을 비교해, 좋아졌을 때만 새 모델로 바꿉니다. 이전 모델로 되돌릴 수 있습니다.
보관 문제	7일 자동 삭제(Q9)를 그대로 적용하면 학습 데이터도 사라집니다. 교정한 문장만 학습용으로 따로 보관해도 되는지 12장 A3에서 확인합니다.

06 게임 연동 (발로란트 · 리그 오브 레전드)

6-1. 발로란트 데이터를 얻는 세 가지 경로

	A. Riot 공식 API (VAL-MATCH-V1)	**B. Riot 클라이언트 로컬 API** (게임 PC에서 실행 중인 클라이언트)	**C. 서드파티 API** (예: HenrikDev)
발급 조건	어려움 프로덕션 키만 가능. 개인 키 신청 불가. 공개 서비스 + RSO 로그인 + 동의 절차 필수. 심사 1~3주 이상, 승인 불확실	불필요 게임 PC에서 로그인된 클라이언트의 인증 정보를 그대로 사용	보통 해당 서비스 키 발급 필요
정책 상태	공식 지원	비공식(문서화되지 않음). 읽기 전용 사용은 다수 트래커 앱이 쓰는 방식이지만 Riot이 보장하지 않음	비공식 제3자 서비스
경기 시작/종료 감지	× (종료 후 조회만)	○ 실시간	× (종료 후)
킬·데스·어시 (시각·위치·무기)	○ 경기 종료 후	○ 경기 종료 후 (경기 상세 조회)	○ 경기 종료 후

라운드 결과·경제·점수	○	○ (경기 중 점수는 실시간)	○
인게임 채팅	×	○ 이 PC 플레이어가 보는 채팅(팀·전체·파티)	×
키·인증 관리	앱 키 + 각 플레이어 RSO 로그인	없음 (클라이언트 실행 중일 때만 동작)	서비스 키

Riot 개발자 정책 요약 (developer.riotgames.com/docs/valorant)
· Personal Key 신청은 지원하지 않음 · 승인 불가 예시: "공개되지 않고 개인용으로만 설계된 앱", 경기 전 상대 전적 조회(스카우팅), "즉시 플레이에 영향을 주는 실시간 데이터 오버레이" · 모든 VALORANT 앱은 사용자 opt-in(RSO) 필수

결정 (Q1): B로 구현하고, A가 필요해지면 그때 공식 API 키를 요청
요청하신 "인게임 채팅 · 실시간 경기 상태"는 공식 API에 아예 없는 데이터라 B 경로로 구현합니다. 게임 연동부를 "데이터 출처 어댑터"로 나눠 두었기 때문에, 나중에 공식 API가 꼭 필요한 기능이 생기면 키 발급을 따로 요청드리고 A 어댑터를 붙입니다. 이 프로그램은 경기 중 플레이에 영향을 주는 정보를 띄우지 않고(오버레이 없음) 기록·복기용이므로, 데이터 사용 성격 자체는 Riot 정책의 "복기·학습" 범주에 해당합니다.

6-2. 수집 항목

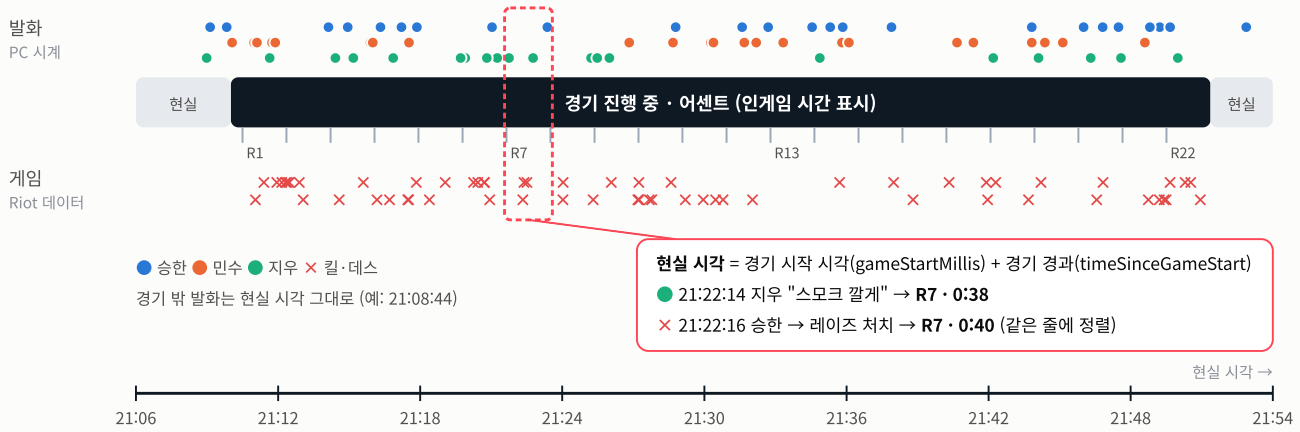
구분	항목
경기 정보	경기 ID, 맵, 모드(경쟁·일반·스파이크 돌격·사설 등), 시작 시각, 길이, 최종 점수, 승패
플레이어	10명의 Riot ID, 요원, 팀, 티어, K/D/A, 점수(ACS), 데미지, 헤드샷 비율
라운드	라운드 번호, 승리 팀·방식(처치/설치 폭발/해체/시간 초과), 스파이크 설치·해체 시각과 위치, 라운드별 경제(구매·잔액)
킬 이벤트	시각(경기 시작 기준·라운드 시작 기준), 킬러, 피해자, 어시스트, 무기·스킬, 피해자 위치와 당시 전원 위치
채팅	시각, 보낸 사람, 채널(팀/전체/파티), 내용

6-3. 시간 동기화 · 인게임 시간 표시

- 경기 데이터에는 **경기 시작 시각(Unix ms)**과 이벤트별 **경기 시작 후 경과 시간**이 있어, 둘을 더하면 현실 시각이 나옵니다. 음성 발화도 현실 시각으로 저장하므로 하나의 시간축으로 정렬할 수 있습니다.
- 로컬 API가 감지한 실제 경기 시작 시점과 비교해 PC 시계와 서버 시계의 차이를 자동 보정합니다.
- Q5 답변(c)에 따라 경기 중 발화는 `R7 · 0:42`(7라운드, 라운드 시작 후 42초)와 경기 경과 시간 `12:18`을 함께 표시합니다. 구매 단계는 `R7 · 구매`로 표시하고, 마우스를 올리면 현실 시각도 보여 줍니다.
- 킬·라운드 상세는 경기가 끝난 뒤 수 분 안에 조회되므로, **경기 중에는 발화와 채팅만 실시간으로 보이고, 경기 종료 후 킬 피드·라운드 정보가 채워지는 애니메이션**으로 표시합니다.

시간 동기화 도식

발화(위)와 경기 이벤트(아래)를 하나의 현실 시간축에 놓고, 경기 구간 안의 항목만 인게임 시간으로 바꿔 표시합니다. 점선 상자는 7라운드입니다.



6-4. 디스코드 유저 ↔ Riot 계정 연결

설정에서 디스코드 유저와 Riot ID를 짝지어 두면, 그 사람의 발화 옆에 같은 순간의 킬·데스가 함께 표시되고 유저 페이지에서 "말한 내용 + 전적"을 같이 볼 수 있습니다. 경기 상세에는 같은 경기에 있던 10명 전원이 포함되므로, 게임 PC 한 대만으로 함께 플레이한 친구들의 기록까지 연결됩니다. 발로란트와 롤은 같은 Riot 계정을 쓰므로 한 번 연결하면 두 게임에 모두 적용됩니다.

6-5. 리그 오브 레전드 연동 (Q11 답변으로 추가)

롤도 Q1 답변과 같은 원칙(게임 PC 로컬 API 우선, 공식 API는 필요할 때 요청)으로 붙입니다. 롤은 게임 PC에서 쓸 수 있는 로컬 API가 두 가지 있어 발로란트보다 데이터 얻기가 수월합니다.

경로	얻는 데이터	비고
게임 클라이언트 API 게임 중 `127.0.0.1:2999`	현재 게임 시간, 10명의 챔피언·KDA·CS·아이템, 이벤트(킬, 첫 킬, 멀티킬, 에이스, 드래곤·전령·바론, 포탑·억제기 파괴, 게임 종료)와 이벤트 시각	Riot 개발자 문서에 소개된 로컬 API. 키 불필요
롤 클라이언트(LCU) API	경기 시작·종료 감지, 큐 종류, 경기 결과, 경기 기록 상세(타임라인 포함 여부는 M0에서 확인)	비공식, 읽기 전용으로만 사용
공식 API (필요할 때만) Match-V5 + 타임라인	킬 위치 좌표, 분당 골드·경험치	롤은 개인용 키 신청이 가능. 필요해지면 그때 요청

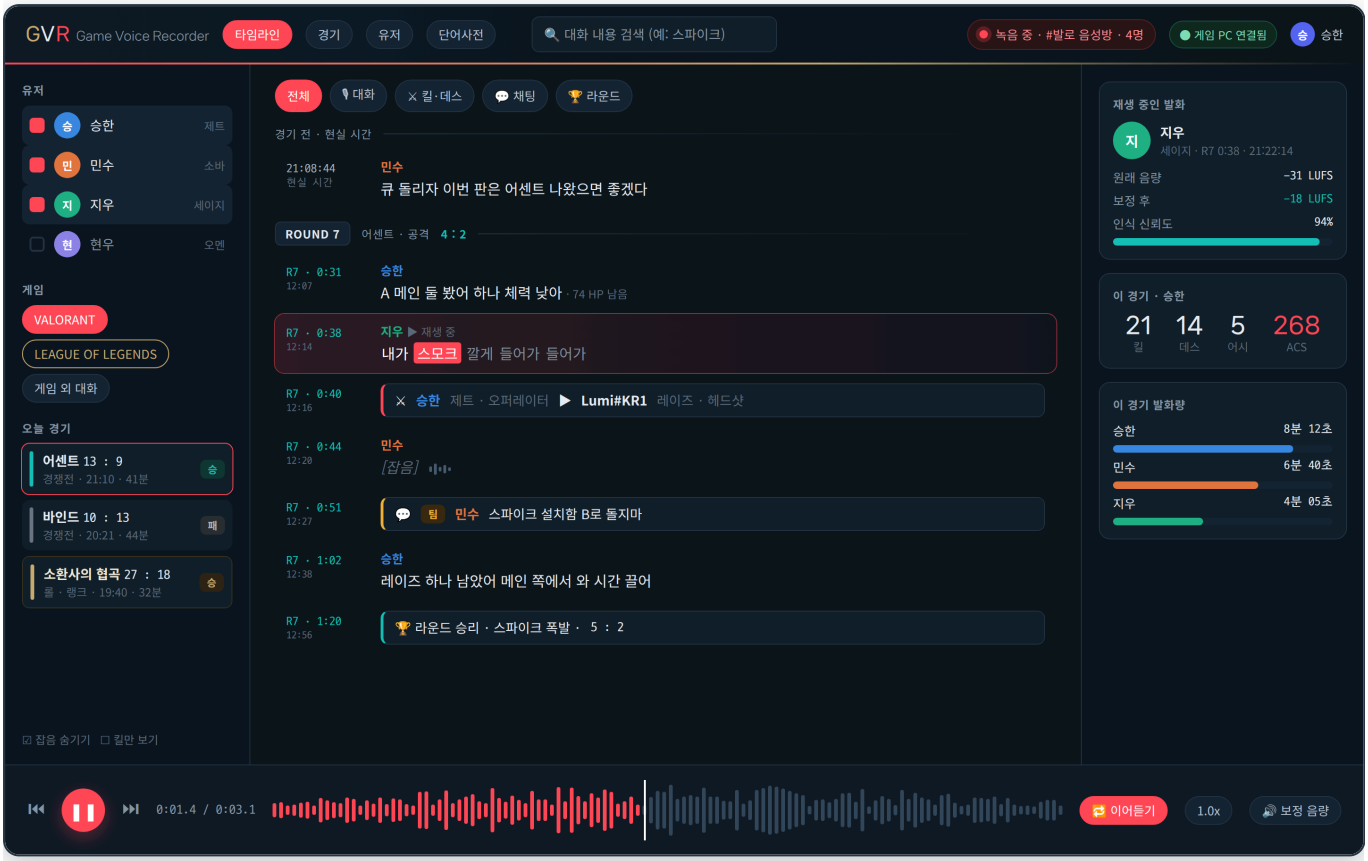
- **시간 표시**: 롤은 라운드가 없어 게임 시간 하나(21:14)로 표시합니다. 게임 클라이언트 API가 주는 게임 시간과 PC 시계를 비교해 현실 시각과 맞춥니다.
- **킬 위치 · 골드 그래프**: 게임 중 이벤트에는 위치가 없어서, 경기가 끝난 뒤 경기 기록(타임라인)으로 채웁니다. LCU에서 타임라인을 받을 수 없으면 이 두 기능만 공식 API 키가 필요하므로, 그때 키 발급을 요청드립니다.
- **인게임 채팅**: 롤 게임 안 채팅은 공개된 로컬 API로 받을 수 없어 1차 버전에서는 뺍니다. 로비·게임 후 채팅을 클라이언트 채팅으로 받을 수 있는지는 M0에서 확인합니다(12장 A5).



화면 예시 · 롤 경기 상세 위쪽은 점수와 골드 차이 그래프(드래곤·포탑·바론 표시), 왼쪽은 협곡 지도의 킬·오브젝트 위치, 가운데는 게임 시간 기준 대화·킬·오브젝트, 오른쪽은 스코어보드와 발화량입니다. 롤 화면은 골드·헥스테크 색을 씁니다. 시안 · 지도는 도식, 데이터는 예시

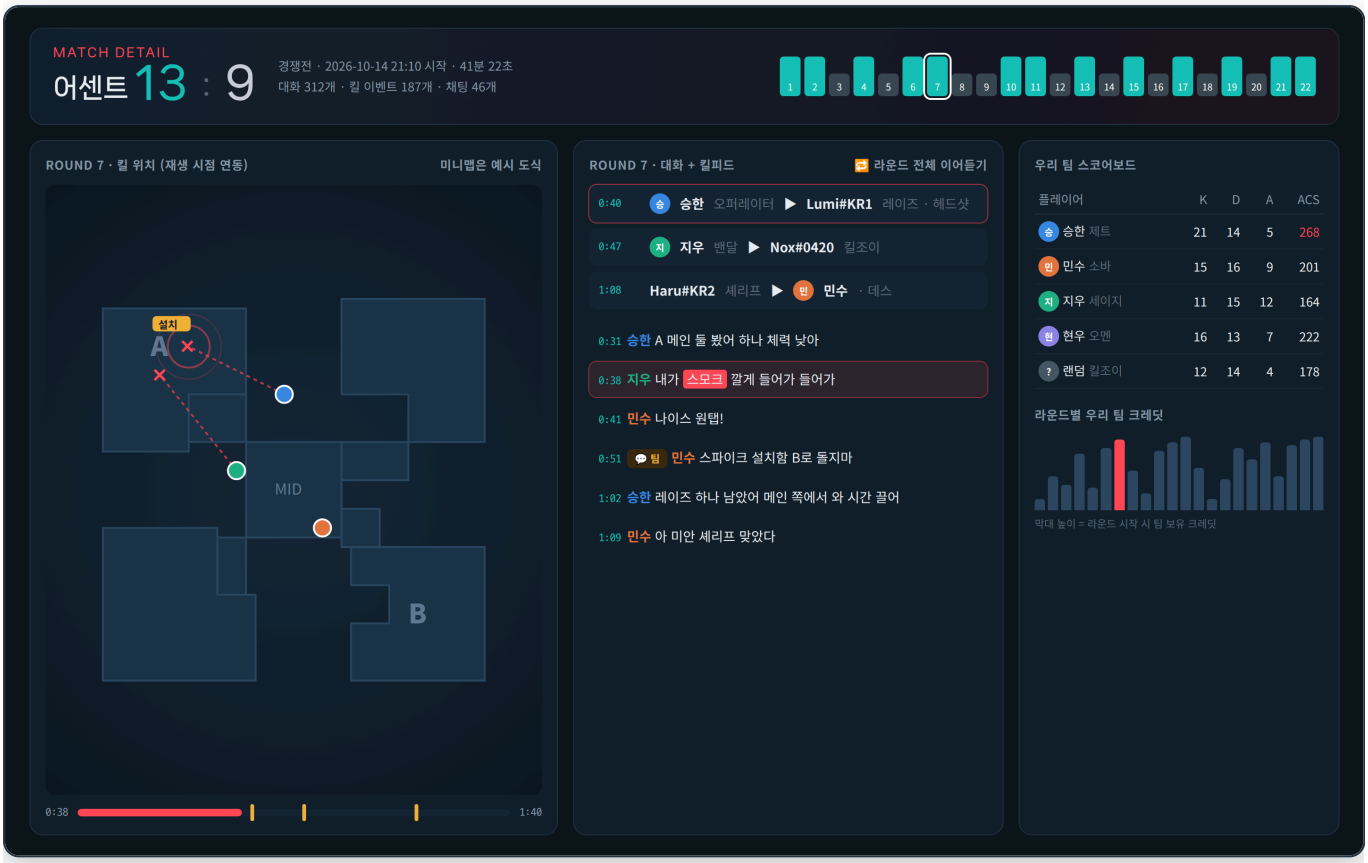
07 사이트 기능 · 접속 · UI/UX 콘셉트

7-1. 메인 화면 (타임라인)



화면 예시 · 타임라인 왼쪽은 유저·게임(발로란트/롤/게임 외 대화)·경기 필터, 가운데는 시간순 대화·킬·채팅·라운드 결과입니다. 시간 칸에는 라운드 시간(R7 · 0:38)과 그 아래 경기 경과 시간(12:14)이 함께 나옵니다. 오른쪽 위에는 게임 PC 연결 상태와 디스코드 로그인 계정이 보입니다. 아래 플레이어에서 파형을 보며 들을 수 있고, 지금 재생 중인 단어("스모크")가 강조됩니다. 시안 · 이름과 데이터는 예시

7-2. 경기 상세 화면 (발로란트)



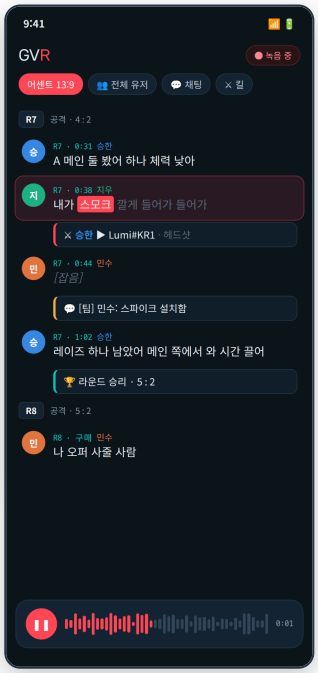
화면 예시 · 경기 상세 위쪽 라운드 띠(초록 = 승리)에서 라운드를 고르면 그 라운드의 킬 위치(미니맵), 대화와 킬피드, 스코어보드, 라운드별 크레딧이 함께 바뀝니다. 재생 위치에 맞춰 미니맵의 킬 표시가 움직입니다. 미니맵은 실제 맵이 아닌 도식

롤 경기 상세 화면은 6-5에 있습니다. 같은 구성(점수 → 지도 → 대화·이벤트 → 스코어보드)을 쓰고 색 테마만 게임에 맞춰 바뀝니다.

7-3. 디스코드 녹음 안내 (TTS) · 휴대폰 화면



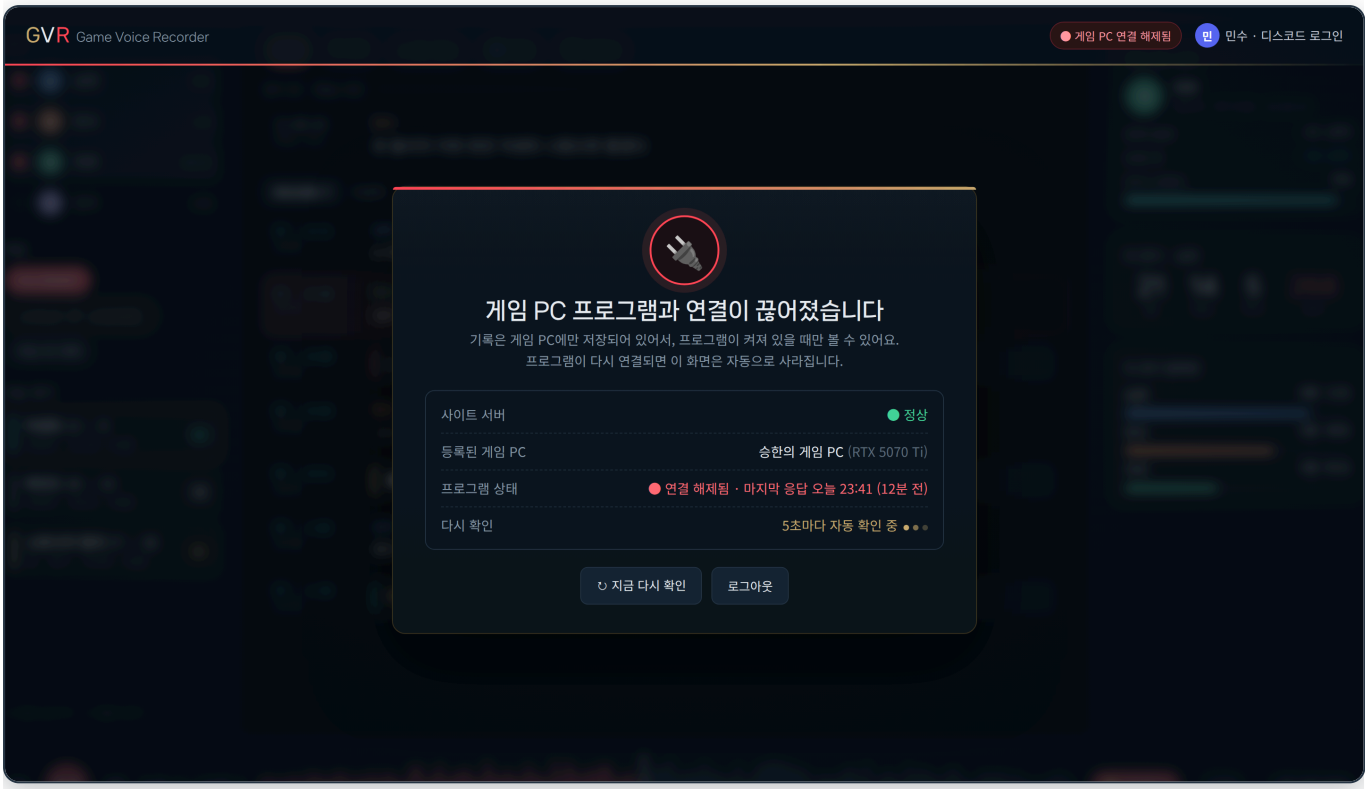
디스코드 녹음 안내 봇이 음성채널에 들어가면 채팅 메시지 없이 TTS 음성으로 녹음 사실을 알립니다. 녹음을 원하지 않는 사람은 /녹음제외, 다시 참여하려면 /녹음참여를 입력합니다. 명령 결과는 본인에게만 보입니다.



휴대폰 화면 같은 사이트를 휴대폰에서도 볼 수 있습니다. 필터는 칩으로 접고, 플레이어는 아래에 고정합니다.

- **안내 시점**: 봇이 음성채널에 들어갈 때, 그리고 녹음 중에 새 사람이 들어올 때 다시 재생합니다(설정에서 끌 수 있음). 안내 음성은 미리 합성해 두고, 안내 문구를 바꾸면 다시 합성합니다.
- **제외 처리**: /녹음제외를 쓴 사람의 음성은 받는 즉시 버리고, 사이트 유저 목록에도 "녹음 제외"로 표시합니다. 제외 상태는 다음 세션에도 유지됩니다.

7-4. 접속 · 디스코드 로그인 · 연결 잠금



화면 예시 · 게임 PC 연결 해제 친구가 디스코드로 로그인해 들어왔지만 게임 PC 프로그램이 꺼져 있는 상태입니다. 사이트 뒤 화면을 흐리게 가리고, 사이트 서버 상태·등록된 게임 PC·마지막 응답 시각을 보여 줍니다. 프로그램이 다시 연결되면 잠금이 자동으로 풀립니다. 시안

- **로그인**: 처음 들어오면 "디스코드로 로그인" 버튼만 보입니다. 봇이 있는 디스코드 서버의 멤버만 통과합니다.
- **연결 확인**: 사이트 서버는 게임 PC 프로그램의 연결 신호를 계속 확인합니다. 등록된 프로그램이 없거나 신호가 끊기면 모든 화면을 잠금 화면으로 바꾸고, 열려 있던 페이지도 실시간으로 잠깁니다.

7-5. 주요 기능

- **따라 읽기 재생**: 문장을 누르면 그 발화부터 재생되고 재생 중인 단어가 강조됩니다. 단어를 누르면 그 단어 위치로 이동하고, "이어듣기"를 켜면 다음 발화로 자동으로 넘어갑니다. 여러 명이 겹쳐 말한 구간은 실제 순서대로 섞어서 재생하는 "현장 모드"도 제공합니다.
- **필터 · 검색**: 유저(복수 선택), 게임(발로란트/롤/게임 외 대화), 경기, 날짜 범위, 키워드(전문 검색), 이벤트 종류(발화/킬/채팅), 잡음 숨기기. 필터 상태가 URL에 남아 링크로 공유할 수 있습니다.
- **경기 상세**: 발로란트는 라운드별 점수 띠, 라운드를 누르면 그 라운드의 대화·킬·채팅만 보기. 롤은 골드 차이 그래프와 오브젝트 표시. 지도에 킬 위치를 찍고 재생 시점에 맞춰 위치를 애니메이션합니다.
- **유저 페이지**: 말한 시간·횟수, 자주 쓰는 단어, 연결된 Riot 계정의 경기 전적.
- **텍스트 교정**: 잘못 인식된 문장을 바로 고치면 교정 사전과 파인튜닝 학습 데이터에 반영됩니다(F-16).
- **실시간 보기**: 녹음 중이면 새 발화가 WebSocket으로 바로 올라옵니다(STT 처리 중에는 파형만 먼저 표시).

7-6. UI/UX 콘셉트: 발로란트 × 리그 오브 레전드

비주얼

공통 바탕은 짙은 남색. 발로란트 화면은 레드·민트 포인트와 각진 HUD 선, 롤 화면은 골드 테두리와 헥스테크 청록을 씁니다. 머리줄은 레드에서 골드로 이어지는 선으로 두 게임을 묶습니다. 유저마다 고유 색과 요원·챔피언 아이콘.

애니메이션

재생 중인 발화의 파형이 실시간으로 움직임 · 발로란트는 라운드 경계 와이프 전환과 킬 플래시, 롤은 오브젝트 처치 때 골드 빛 번짐 · 경기 종료 후 킬피드가 시간 순으로 "채워지는" 연출 · 잠금 화면이 풀릴 때 열리는 전환.

효과음

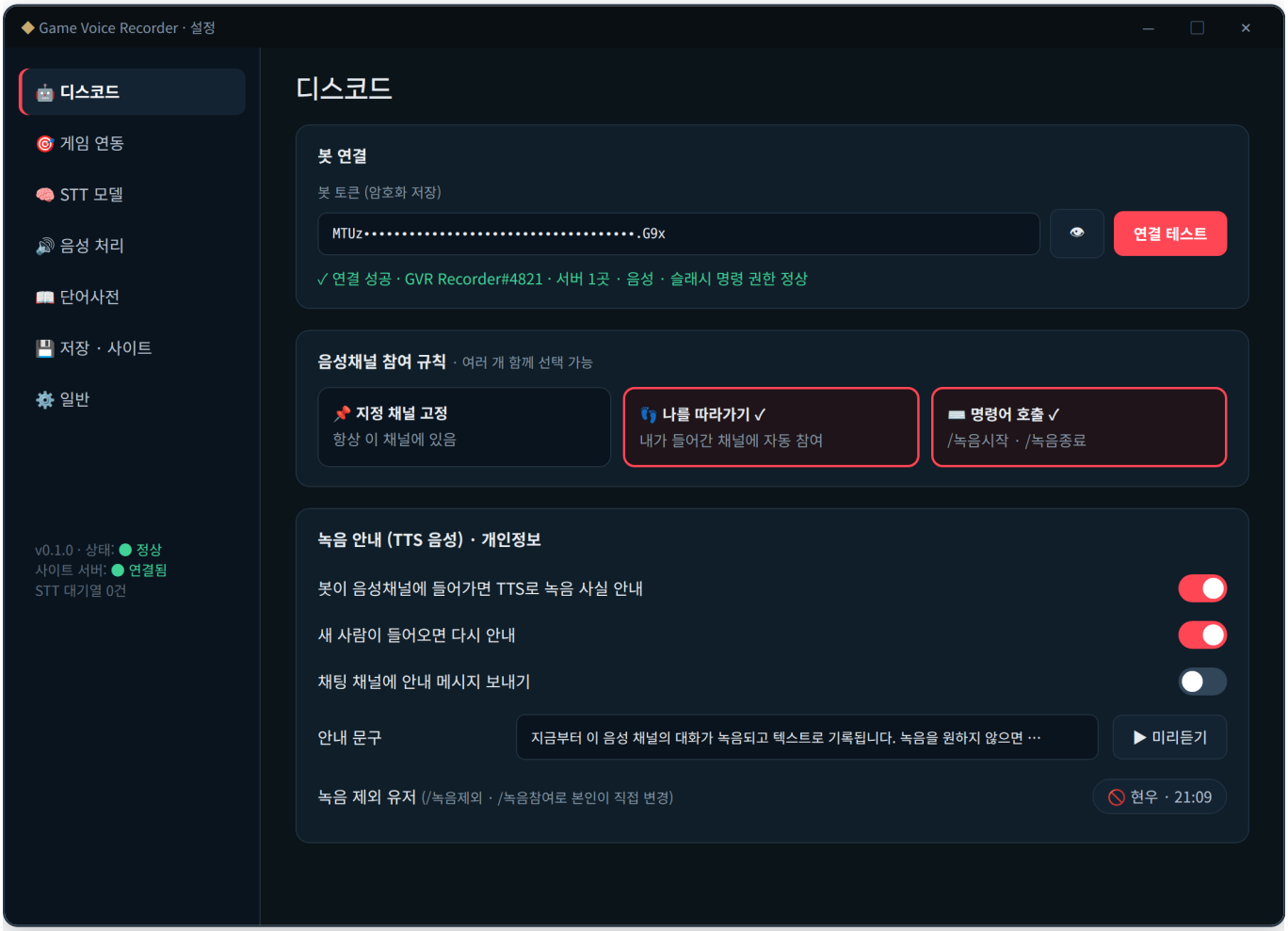
필터·버튼 클릭 틱, 재생 시작 효과음, 킬 이벤트 지나갈 때 짧은 효과음, 라운드 전환음, 새 발화 실시간 알림음. 전체 볼륨·종류별 켜기/끄기. 효과음은 직접 제작하거나 무료 라이선스 음원 사용(게임 원본 음원은 저작권상 사용하지 않음).

편의 · 접근성

단축키(스페이스 재생, ←/→ 이전·다음 발화, / 검색), "모션 줄이기" 설정 시 애니메이션 최소화, 휴대폰 화면 대응, 앱 셸에서는 녹음 상태와 사이트 서버 연결 상태를 트레이 아이콘 색으로 표시.

08 exe 설정 페이지

첫 실행 때는 마법사로 순서대로 안내하고, 이후에는 트레이 아이콘 → 설정 창에서 언제든 바꿀 수 있습니다. 비밀값은 Windows DPAPI로 암호화해 앱 폴더에 저장합니다.



화면 예시 · 설정 창 (디스코드 탭) 봇 토큰을 입력하고 연결을 테스트합니다. 참여 규칙은 "나를 따라가기"와 "명령어 호출"을 함께 켜고, 녹음 안내는 TTS 음성으로 합니다. 제외 유저는 본인이 /녹음제외로 직접 바꿉니다. 시안

탭	항목
디스코드	봇 토큰(입력 후 연결 테스트), 대상 서버 선택, 참여 규칙(기본: 나를 따라가기 + /녹음시작·/녹음종료), TTS 녹음 안내(문구·새 사람 입장 시 재안내), 녹음 제외 유저 확인
게임 연동	사용할 게임(발로란트 · 리그 오브 레전드), 게임별 연결 상태 표시, 디스코드 유저 ↔ Riot ID 매핑, 공식 API 키 칸(필요해질 때만 사용)
STT 모델	감지된 GPU·VRAM·드라이버 표시, 실시간 모델·정밀 모델 선택, 다운로드 폴더, 다운로드·삭제·교체, 파인튜닝(지금 학습 / 매일 새벽 자동 / 이전 모델로 되돌리기)
음성 처리	목표 음량, 노이즈 억제 켜기/끄기, 잡음 판정 민감도, 발화 분할 무음 길이 (인식 언어는 한국어 고정)
단어사전	기본 사전 켜기/끄기(게임·일상·대학생·20대 등 묶음별), 사용자 단어 추가, 교정 규칙 편집, 사이트 교정 내역 반영 여부
저장 · 사이트	녹음 저장 폴더, 보관 기간(기본 7일, 지나면 자동 삭제), 사용 용량 표시, 사이트 서버 주소·등록 코드·연결 상태, 로그인 허용 디스코드 서버
일반	윈도우 시작 시 자동 실행(사이트를 계속 열어 두려면 권장), 효과음·테마, 로그 보기, 업데이트 확인

09 데이터 저장 구조 · 용량

테이블	주요 컬럼
users	디스코드 ID, 표시 이름, 색, 아바타, 녹음 제외 여부(/녹음제외), 게인 프로필
game_accounts	유저 ID, Riot PUUID, Riot ID (발로란트·롤 공통)
sessions	음성채널 세션 시작·종료, 서버·채널
utterances	화자, 시작·종료 시각(ms), 음성 파일 경로, 측정 음량, 텍스트, 단어별 시각(JSON), 잡음 여부, 신뢰도, 모델 이름, 경기 ID·라운드, 교정 여부
matches / rounds	게임 종류, 경기 정보, (발로란트) 라운드 결과·경제·스파이크 이벤트
game_events	종류(킬·채팅·오브젝트 등), 현실 시각, 경기 시간, 관련 플레이어, 상세(JSON)
dictionary	단어, 묶음(발로란트·롤·대학생 등), 교정 규칙, 출처(기본/사용자/사이트 교정)
training_samples	교정한 문장, 음성 조각 경로, 사용 여부 (보관 기간은 A3 답변에 따름)
사이트 서버 devices	등록된 게임 PC, 기기 키 해시, 마지막 연결 시각 (사이트 서버에는 기록 내용을 저장하지 않음)

≈14 MB
말한 시간 1시간
(Opus 32kbps)

≈60 MB
하루 사용량
4명 × 각자 1시간 발화

≈0.4 GB
최대 저장 용량
7일 보관 기준

7 일
지나면 자동 삭제
(매일 새벽 정리)

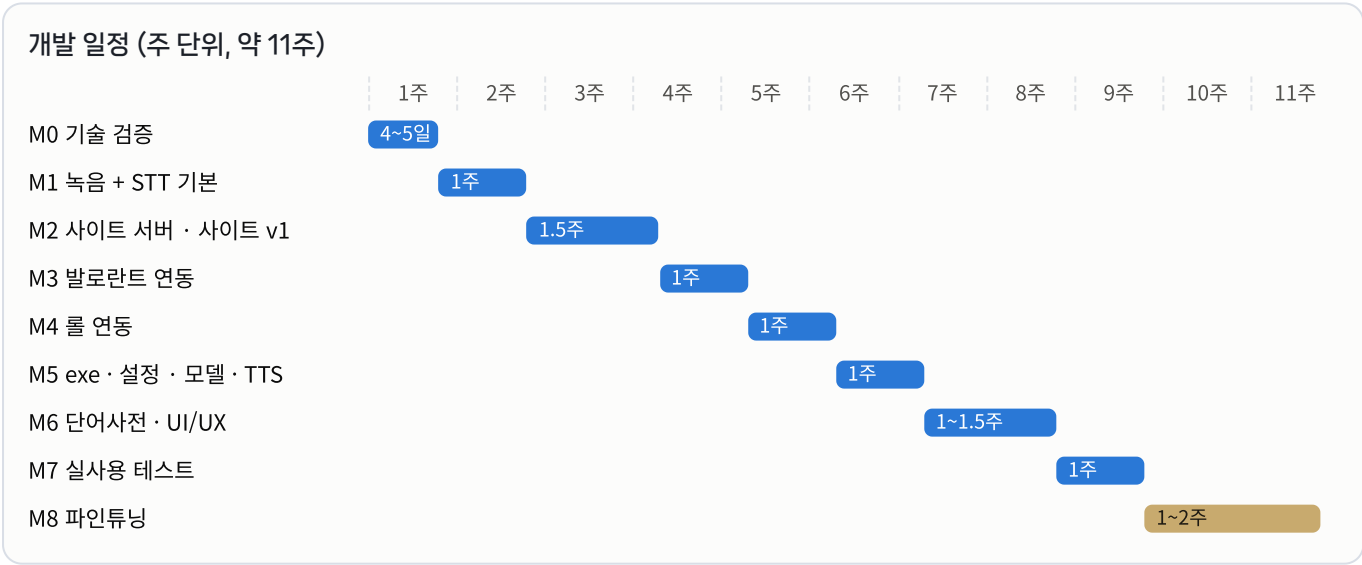
가정: 4명이 하루 3시간 함께 게임하고, 각자 실제로 말하는 시간이 하루 1시간 정도. 7일 보관이라 용량은 약 0.4GB 이상 늘지 않습니다. 모델 파일(약 5GB)은 별도입니다. 실제 사용량은 설정의 "사용 용량"에서 확인할 수 있습니다.

10 기술 스택 · 개발 일정

영역	선택	이유

앱 셸 · 패키징	Electron + electron-builder (portable)	설정 창·트레이·웹 UI 공유, 포터블 exe 지원
코어	Node.js + TypeScript	디스코드 음성 라이브러리 성숙도
디스코드	discord.js, @discordjs/voice (DAVE), 슬래시 명령	음성 수신 + 2026년 필수 E2EE 대응, TTS 안내 재생
사이트 서버	Node.js + Fastify, WebSocket 중계, Docker, HTTPS	로그인·연결 확인·중계만 담당, 기록은 저장하지 않음
로그인	Discord OAuth2 (identify, guilds)	디스코드 서버 멤버만 접속 (Q10)
게임 데이터	발로란트 로컬 API, 롤 게임 클라이언트 API · LCU	키 없이 게임 PC에서 수집 (Q1)
웹 서버 · DB	Fastify, SQLite(better-sqlite3, FTS5)	설치 불필요, 단일 파일 DB, 한국어 전문 검색
웹 UI	React + Vite, Framer Motion, wavesurfer.js, Howler.js	애니메이션 · 파형 · 효과음
STT 워커	Python 3.11, faster-whisper (CTranslate2 4.5 이상, CUDA 12.8 이상), Silero VAD, pyloudnorm, PyInstaller	Blackwell GPU STT · 음량 정규화 · 별도 프로세스 패키징
파인튜닝 · TTS	PyTorch, PEFT(LoRA), 한국어 TTS (엔진은 M0에서 선택)	모델 추가 학습, 안내 음성과 학습용 음성 합성
GPU 감지	NVML (pynvml)	GPU 이름·VRAM·드라이버 확인
빌드 · CI	Gitea Actions + Windows 빌드 환경	exe 자동 빌드 · 릴리스

일정 (답변 반영, 약 11주)



단계	내용	완료 기준	예상
M0	기술 검증	DAVE 환경에서 유저별 음성 수신, RTX 5070 Ti에서 faster-whisper 속도·VRAM 실측(CUDA 12.8), 발로란트·롤 로컬 API로 경기 감지, 롤 타임라인·채팅 가능 여부 확인	4~5일
M1	녹음 + STT 기본	봇 참여(따라가기·명령) → TTS 안내 → 발화 저장 → 음량 정규화 → STT → DB 저장 → 잡음 판정, /녹음제외·/녹음참여	1주

M2	사이트 서버 · 사이트 v1	사이트 서버(Docker), 게임 PC 등록·연결 확인·잠금 화면, 디스코드 로그인, 타임라인·따라 읽기 재생·필터·검색·실시간 갱신	1.5주
M3	발로란트 연동	경기 감지, 경기 상세·킬·라운드·채팅 수집, 시간 동기화, 라운드 시간 + 경기 경과 시간 표시	1주
M4	롤 연동	게임 클라이언트 API 이벤트 수집, LCU 경기 감지·결과, 게임 시간 표시, 롤 경기 상세 화면	1주
M5	exe · 설정 · 모델 관리	포터블 exe, 첫 실행 마법사, GPU 감지·모델 자동 다운로드, 설정 저장(암호화), 7일 자동 삭제	1주
M6	단어사전 · UI/UX 완성	핫워드·교정 사전(발로란트·롤·일상), 사이트 교정 반영, 발로란트×롤 테마·애니메이션·효과음·지도	1~1.5주
M7	실사용 테스트 · 안정화	실제 게임 세션으로 인식률·게임 프레임 영향 측정, 기준값 조정, 버그 수정	1주
M8	파인튜닝	교정 데이터 + TTS 합성 음성으로 LoRA 학습, 오류율 비교 후 모델 교체 (실사용 데이터가 2주 정도 쌓인 뒤 시작)	1~2주

11 위험 요소 · 대응

위험	영향	대응
Riot 공식 API 키 미승인	"공식 API" 경로 불가	로컬 API로 구현(Q1). 공식 API가 꼭 필요한 기능이 생기면 그때 키를 요청하고 어댑터만 추가 (6장)
로컬 API 변경 · 정책	패치 후 일부 데이터 수집 실패, 비공식 경로의 정책 불확실성	읽기 전용만 사용, 메모리 접근·주입 없음, 실패해도 녹음·STT는 계속 동작, 엔드포인트를 한 곳에 모아 빠르게 수정
롤 데이터 한계	게임 안 채팅 수집 불가, 킬 위치는 경기 후 타임라인 필요	채팅은 1차 제외(A5), 타임라인을 LCU로 받지 못하면 롤 개인용 공식 키 요청
RTX 50 시리즈 호환	CUDA·CTranslate2 버전이 맞지 않으면 GPU STT 실패	CUDA 12.8 이상 런타임, CTranslate2 4.5 이상, float16 기본. M0에서 실측하고 필요하면 직접 빌드
사이트 서버 · 연결 끊김	게임 PC가 꺼지거나 네트워크가 끊기면 사이트를 볼 수 없음	의도된 동작(Q2). 잠금 화면으로 상태를 알리고 자동 재연결. 기록은 게임 PC에 있어 유실 없음
7일 삭제와 파인튜닝	학습 데이터가 함께 지워짐	교정한 문장만 학습용으로 따로 보관하는 안을 A3에서 확인
디스코드 음성 수신 변경	음성 수신은 디스코드가 공식 문서화하지 않은 기능	라이브러리 버전 고정 · 업데이트 추적, 수신 실패 감지 시 알림
게임 중 GPU 경합	게임 프레임 저하	VRAM 예산, 낮은 우선순위, 경기 중 경량 모드 / 경기 후 처리
Whisper 환각 · 짧은 발화 오인식	없는 문장 생성	VAD 선처리, 환각 문구 필터, 신뢰도 기준 [잡음] 처리
녹음 동의 · 개인정보	참가자 동의 없는 녹음 문제, 채팅 안내가 없어 안내를 놓칠 수 있음	입장 때와 새 사람이 들어올 때마다 TTS 안내, /녹음제외, 7일 자동 삭제, 디스코드 로그인으로 접근 제한

12 질문 답변 정리 · 추가 확인 사항

12-1. Q1~Q11 답변과 반영 내용

Q1 발로란트 데이터 경로

답변 B로 구현하고, A가 필요하면 공식 API를 달라고 다시 요청할 것

→ 반영: 발로란트·롤 모두 게임 PC 로컬 API로 구현. 공식 API가 필요한 기능이 생기면 그때 키 발급을 요청 (6장)

Q2 사이트를 누가 어디서 보나요?

답변 C. 내 컴퓨터 프로그램과 계속 확인해서, 프로그램 등록이 안 되어 있으면 사이트를 가리고 "프로그램 연결이 해제됨"처럼 표시

→ 반영: 별도 사이트 서버 + 게임 PC가 먼저 연결하는 중계 구조. 등록·연결이 없으면 잠금 화면 (3장, 7-4)

Q3 게임 PC GPU · AMD/Intel 지원

답변 NVIDIA 하나만. 현재 게임 PC GPU는 RTX 5070 Ti

→ 반영: faster-whisper(CUDA)만 사용, 16GB 기준 모델 구성, CUDA 12.8 이상 (5장)

Q4 경기 중 바로 텍스트가 보여야 하나요?

답변 a (경기 중 실시간 변환 + 경기 후 정밀 재변환)

→ 반영: 실시간 large-v3-turbo → 경기 후 large-v3 재변환 (4-4, 5-2)

Q5 인게임 시간 표시 형식

답변 c (둘 다)

→ 반영: 발로란트는 R7 · 0:42 + 경기 경과 12:18 함께 표시, 롤은 게임 시간 (6-3, 6-5)

Q6 단어사전 학습 범위

답변 a + b (핫워드·교정 사전 + 파인튜닝)

→ 반영: 1·2단계는 처음부터, 파인튜닝은 M8에서 게임 PC로 학습 (5-4, 5-5)

Q7 봇이 음성채널에 들어가는 방식

답변 b + c (나를 따라가기 + 슬래시 명령)

→ 반영: 따라가기 + /녹음시작 · /녹음종료 (8장)

Q8 녹음 동의 · 안내

답변 메시지는 보내지 않고, 봇이 음성채널에 들어간 뒤 TTS로 녹음 사실을 알림. 원하지 않는 사람은 /녹음제외로 빠지고 /녹음참여로 다시 참여

→ 반영: TTS 안내(입장 때 + 새 사람 입장 때), /녹음제외 · /녹음참여 (7-3)

Q9 보관 기간

답변 일주일 후 자동 삭제

→ 반영: 7일 지나면 자동 삭제, 최대 약 0.4GB (9장). 삭제 범위는 A2에서 확인

Q10 사이트 로그인 방식

> **답변** c (디스코드 로그인)

→ 반영: 봇이 있는 디스코드 서버의 멤버만 접속 (7-4)

Q11 그 밖의 선호 사항

> **답변** 인식 언어는 한국어 고정 · 디자인은 발로란트 + 리그 오브 레전드 분위기 · 롤도 같이 한 번에 추가

→ 반영: language="ko" 고정, 발로란트×롤 테마, 롤 연동을 1차 범위에 포함 (5장, 6-5, 7-6)

12-2. 추가로 확인할 것

답이 없으면 각 항목의 추천안대로 진행합니다. 다만 A1(서버 주소)과 A4(봇 토큰)는 직접 정하거나 준비해 주셔야 구현을 시작할 수 있습니다.

A1 사이트 서버를 어디에 두고, 어떤 주소로 열까요?

친구들이 밖에서 접속하려면 항상 켜져 있는 서버와 HTTPS 주소가 필요합니다. 서버에는 로그인과 중계만 있어서 가볍습니다.

추천: 지금 쓰고 있는 Docker 서버에 컨테이너로 올리고, 서브도메인(예: gvr.tkrmagid.kr)을 연결

A2 7일 자동 삭제는 어디까지 지울까요?

(a) 음성 파일만 · (b) 음성 + 대화 텍스트 · (c) 음성 + 텍스트 + 경기 기록(킬·점수 등)까지 모두

추천: (b). 경기 전적 숫자는 용량이 거의 없어 남겨 두고, 목소리와 대화 내용만 7일 뒤 삭제

A3 파인튜닝용 학습 데이터는 7일 넘게 보관해도 될까요?

파인튜닝(Q6-b)에는 고친 문장과 그 음성 조각이 필요한데, 7일 뒤 지우면 학습 데이터가 쌓이지 않습니다.

추천: 사이트에서 직접 고친 문장만 학습용으로 따로 보관(/녹음제외 유저는 제외, 설정에서 언제든 전체 삭제 가능)

A4 봇 토큰은 이미 있나요? 사용할 디스코드 서버는 몇 개인가요?

Q7에서 아직 답을 받지 못한 부분입니다. 봇 하나는 서버마다 음성채널 하나에만 들어갈 수 있고, 로그인 허용 범위도 이 서버 기준으로 정해집니다.

없으면 디스코드 개발자 포털에서 봇을 새로 만들어 토큰을 설정 화면에 넣으면 됩니다

A5 롤 인게임 채팅은 1차 버전에서 빼도 될까요?

롤 게임 안 채팅은 공개된 로컬 API로 받을 수 없습니다. 발로란트 채팅은 그대로 수집합니다.

추천: 1차는 제외, 로비·게임 후 채팅 수집 가능 여부만 M0에서 확인