

확장형 멀티퍼포스 디스코드 봇
'아카리(Akari)'
초기 인프라 구축

문서 번호	DEV-2025-1226-01	작성 일자	2025. 12. 26.
작 성 자	김태랑	검 토 자	-
제 목	차세대 AI 기반 확장형 봇 '아카리' 서비스 기반 구축 완료 보고		

1. 개요

1.1. 추진 배경 및 목적

- 확장형 통합 플랫폼 구축: 단순 편의 기능을 넘어 게임, 관리, 유틸리티 등 다양한 기능을 수행할 수 있는 멀티퍼포스(Multi-purpose) 봇의 초석 마련.
- 음성 소통의 현대화 (Phase 1): 첫 번째 핵심 모듈로 TTS(Text-to-Speech) 시스템을 우선 탑재하여, 청각적 소통이 불가능한 환경의 사용자 경험을 개선함.
- 최신 AI 개발 트렌드 적용: 생성형 AI 기술을 개발 프로세스 전반에 적극 도입하여 코드 효율성을 극대화하고 최신 기술 트렌드를 선도함.

1.2. 프로젝트 범위

- 기간: 2025. 12. 25. ~ 2025. 12. 26. (2일간)
- 대상: 디스코드 서버 내 음성 및 텍스트 채널을 이용하는 모든 사용자.

최종 산출물:

- 모듈형 설계를 적용한 Python 기반 봇 코어 (Github)
- Render 클라우드 기반 24시간 호스팅 환경
- AI 협업 기반의 개발 문서 및 가이드라인

2. 개발 환경 및 기술 명세 (Technical Specifications)

2.1. 개발 방법론: AI-Assisted Development

본 프로젝트는 최신 AI 기술을 활용한 애자일(Agile) 방식으로 진행됨.

- AI 협업 코딩: LLM(Large Language Model)을 활용하여 최적화된 알고리즘 설계 및 트러블슈팅 시간 단축.
- 코드 품질 향상: AI 기반의 코드 리뷰 및 리팩토링을 통해 유지보수성과 확장성이 용이한 구조 확보.

2.2. 소프트웨어 및 라이브러리

구분	상세 내용	비고
언어 (Language)	Python 3.10.0	모듈 확장에 최적화된 버전
핵심 라이브러리	discord.py (2.x)	비동기 기반의 고성능 API 처리
기반 모듈 (Core)	Module System	기능(Cogs) 단위의 독립적 로딩 구조 설계
1단계 모듈 (TTS)	gTTS & FFmpeg	음성 합성 및 스트리밍 처리
인프라 (Infra)	Render & Flask	24시간 무중단 클라우드 환경

2.3. 인프라 및 배포 환경

- 호스팅 서버: Render (PaaS 기반 Web Service)
 - 선택 사유: Docker 컨테이너 기반의 안정적 배포 지원 및 Git 연동의 편의성.
- 운영체제(OS): Linux (Render 컨테이너 환경)
- 모니터링: UptimeRobot
 - 설정: 5분 간격 HTTP Request 전송을 통한 24시간 가동 유지(Keep-Alive).

3. 상세 구현 내용

3.1. 확장성을 고려한 코어 설계 (Core Architecture)

가. 모듈형(Cogs) 구조 도입

- 현재 구현된 TTS 기능 외에 향후 추가될 미니게임, 서버 관리 기능 등이 서로 간섭하지 않고 독립적으로 작동하도록 설계함.
- Extension Load 방식을 통해 붓을 끼지 않고도 새로운 기능을 실시간으로 추가/수정할 수 있는 유연성 확보.

나. 슬래시 커맨드(Slash Command) 체계 구축

- /join, /leave 등 직관적인 UI를 적용하되, 향후 /game, /rank 등으로 명령어를 확장하기 용이한 트리 구조 적용.

3.2. 1단계 핵심 기능: TTS 및 오디오 시스템

- 크로스 플랫폼 지원: 개발(Windows) 및 배포(Linux) 환경을 자동 감지하여 FFmpeg 경로를 동적으로 할당, 어떤 환경에서도 오디오 기능이 작동하도록 구현.
- 이벤트 리스너 최적화: 단순 채팅 읽기를 넘어, 향후 AI 챗봇 기능을 탑재할 수 있도록 메시지 처리 로직(on_message)을 분리 설계함.

3.3. 서버 구축 및 24시간 가동 체계

- PaaS 기반 클라우드 이관: 로컬 구동의 한계를 극복하고 언제 어디서든 접근 가능한 상시 대기 서버 구축.
- 무중단 모니터링: Flask 웹 서버와 UptimeRobot을 연동하여, 추가 비용 없이 24시간 깨어있는 시스템을 완성함.

4. 문제 해결(Troubleshooting) 및 조치 사항

구분	주요 성과 및 해결 내역
개발 효율화	AI 활용: 라이브러리 충돌 및 의존성 문제를 AI 분석을 통해 즉각 해결, 개발 기간을 획기적으로 단축함.
인프라 최적화	FFmpeg 자동 배포:build.sh 스크립트를 통해 클라우드 환경에서 미디어 엔진이 자동 설치되도록 파이프라인 구축.
권한 관리	Intents 설정: 보안 정책을 준수하면서도 메시지를 원활히 수집할 수 있도록 권한 체계 확립.

5. 결론 및 향후 계획

5.1. 종합 평가

- 본 프로젝트는 단순한 기능 구현을 넘어, 미래 확장이 가능한 봇의 '뼈대(Skeleton)'를 성공적으로 구축했다는 데에 의의가 있음.
- 특히 AI 기술을 활용한 개발 프로세스를 정립함으로써, 향후 기능 추가 시에도 빠르고 정확한 업데이트가 가능한 환경을 마련함.
- TTS 기능은 본 플랫폼의 첫 번째 서비스일 뿐이며, 이를 통해 인프라의 안정성(24시간 가동)을 검증 완료함.

5.2. 향후 확장 로드맵 (Roadmap)

현재 구축된 인프라를 바탕으로 다음과 같은 기능 확장을 단계적으로 추진할 계획임.

1. 엔터테인먼트 모듈: 주사위, 사다리 타기, RPG형 미니게임 등 사용자 참여형 콘텐츠 탑재.
2. AI 인텔리전스 모듈: ChatGPT 등 LLM API를 연동하여 사용자와 자연스러운 대화가 가능한 AI 비서 기능 추가.
3. 서버 유틸리티: 유저 관리, 자동 경고 시스템 등 관리자 편의 기능 강화.

위와 같이 확장형 멀티퍼포스 봇 '아카리'의 기반 구축 완료를 보고합니다.