**NewsReader**

작성자: \_\_\_\_\_\_\_\_이준혁\_\_\_\_\_\_\_\_  
소속: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_학생\_\_\_\_\_\_  
제출일: \_\_\_\_2025.09.09\_\_\_\_\_\_\_\_

# Table of Contents

1. Introduction

1.1 Purpose

1.2 Scope

1.3 Definitions, acronyms, and abbreviations

1.4 References

1.5 Overview

2. Overall description

2.1 Product perspective

2.2 Product functions

2.3 User Characteristics

2.4 Constraints

2.5 Assumptions and dependencies

3. Specific requirements

3.1 External Interfaces

3.1.1 User Interfaces

3.1.2 Hardware Interfaces

3.1.3 Software Interfaces

3.1.4 Communication Interfaces

3.2 Functional requirements

3.3 Performance requirements

3.4 Design constraints

3.5 Software system attributes

# Introduction

* 1. Purpose
* 본 문서의 목적은 외신을 포함한 기사들의 번역 및 음성 응답 서비스 NewsReader에 대한 요구하상을 명확히 정의하는 것이다. 이 문서는 개발 팀, 이해 관계자 및 유지보수 팀이 시스템이 주요 기능, 제약 사항 및 개발 범위를 이해하고, 이를 기반으로 효율적이고 안정적인 기사 수집 및 번역 시스템을 구축 및 관리하는 데 필요한 정보를 제공한다.
  1. Scope
* 본 시스템의 정식 명칭은 NewsReader이다. NewsReader은 API Server를 통해 다양한 기사를 수집하고, 이를 클라이언트에 번역 및 기사 전송 등의 기능을 제공한다. 주요 구성 요소는 API Server, Client이며, 각 구성 요소는 독립적으로 동작한다.
* 이 SRS는 NewsReader의 소프트웨어 요구사항에 대한 전반적인 설명을 포함하며 설치 환경, 네트워크 구성, 보안 인증 방식 등의 인프라 설계 및 배포 세부사항은 본 문서 범위에 포함되지 않는다.
  1. Definitions, acronyms, and abbreviations
* NewsReader: 본 프로젝트의 정식 명칭이자 서비스 이름
* API Server: 기사 수집 및 음성 회독 서비스를 제공하는 API 서버
  1. References
* IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications
* NewsReader 프로젝트 문서:<https://github.com/Sadturtleman/NewsReader>

# Overall Description

2.1 Product Perspective

* NewsReader은 뉴스 기사를 번역 및 음성 지원하는 서비스로 해외의 기사에 대한 단절을 줄이는 것을 목표로 한다.
* 이 시스템은 API Server와 Client로 구성되며, 각 모듈은 독립적으로 동작한다.
* NewsReader은 다양한 OS(Window, Web, Android)에서 지원되며, HTTP 프로토콜로 동작가능한 API Server를 제공한다.

2.2 Product Functions

* 뉴스 기사 수집: 미리 지정한 신문사의 기사를 수집, 가공 및 저장
* 음성 회독: 문장의 음성 파일 제공
* 기사 번역: 문장의 지정된 언어로의 번역
* 데이터 제공: 요구된 기사와 음성, 유저 정보 제공
* 로그인: 자신이 지정한 아이디로 로그인 가능
* 알림 기능: 자신이 설정한 메신저와 연결되어 알림을 제공

2.3 User Characteristics

* NewsReader은 프로그래밍 지식이 부족한 사용자도 쉽게 사용할 수 있도록 전문적인 지식을 드러내서는 안 되며, 직관적인 UI, UX를 제공한다.
* 사용자는 다양한 연령대와 직업으로 구성되며, 다국어 UI와 장애인을 위한 음성 서비스를 제공한다.

2.4 Constraints

* 기사 및 음성 데이터 수집은 API 기반이며, 키가 정상적으로 등록이 되 있어야 한다.
* 시스템은 Multi-Tenant 환경을 지원해야 하며, 각 Tenant간에 독립적인 수행을 보장해야 한다.
* 시스템의 가용성은 프로젝트 운영기간동안 95% 이상을 유지해야 한다.
* 시스템은 과도한 초기사항을 요구해서는 안 된다.

2.5 Assumptions and Dependencies

* 외부 시스템과 연결에 필요한 API Key 등은 안정적으로 구성이 되어있다고 가정한다.

# Specific Requirements

3.1 External Interfaces

3.1.1 User Interfaces

텍스트, 스크린샷, 디자인이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

3.1.2 Hardware Interfaces

* + 해당 사항 없음

3.1.3 Software Interfaces

* + 기사 수집 인터페이스 (RSS -> API Server)
    - RSS로 수집한 데이터를 API Server에 저장한다.
    - 프로토콜: HTTP POST
  + 음성 인터페이스 (Voice API <-> API Server)
    - 요청한 문장에 대한 음성 파일을 제공한다.
    - 프로토콜: HTTP GET
  + 번역 인터페이스 (Translator API <-> API Server)
    - 요청한 문장과 언어에 대한 번역을 제공한다.
    - 프로토콜: HTTP GET
  + 사용자 인증 및 설정 관리 인터페이스(Frontend <-> API Server)
    - 로그인, 회원가입 등 사용자 인증 기능을 제공한다.
    - 설정 등록 기능을 제공한다.
    - 즐겨찾기 기능을 제공한다.
    - 프로토콜: HTTP REST API
    - 보안: JWT 기반 인증 및 HTTPS
  + 데이터 제공 인터페이스(API Server -> Frontend)
    - 사용자가 요청한 기사, 음성, 번역에 대한 파일을 제공한다.
    - 프로토콜: HTTP GET

3.1.4 Communication interfaces

|  |  |  |
| --- | --- | --- |
| 송신 주체 | 수신 주체 | 설명 |
| Voice API | API Server | 요청된 문장에 대한 음성 파일을 제공한다. |
| Translation API | API Server | 요청된 문장과 언어에 대한 번역을 제공한다. |
| API Server | Frontend | 요청된 데이터에 대한 제공을 처리하며, 사용자 인증을 관리한다. |

3.2 Functional Requirements

3.2.1 API Server

3.2.1.1 Initialization

3.2.1.1.1 서버 실행(FR-1.1.1)

* 시스템은 실행 시 Frontend와 통신하기 위한 서버를 시작해야 한다.
* 이 서버는 Frontend와 API와의 통신 요청을 수신 및 응답한다.
* 프로그램 종료 시까지 대기 상태를 유지한다.

3.2.1.1.2 연결 유효성 검사(FR-1.1.2)

* 시스템은 시작 시 API와 Frontend간의 통신이 원활한지 검사해야 한다.
* 통신이 유효할 경우 대기상태를 유지한다.
* 통신이 원활하지 않을 경우 일정 기간에 거쳐 반복하며, 그 장애를 로그에 기록해야 한다.

3.2.1.2 Management

3.2.1.2.1 음성 데이터 수신(FR-1.2.1)

* 시스템은 Voice API에 문장을 보내어 음성 데이터를 수신 받아야 한다.
* 받지 못할 경우 일정 기간에 거쳐 반복하며, 그 장애를 로그에 기록해야 한다.

3.2.1.2.2 음성 데이터 송신(FR-1.2.2)

* 시스템은 요청된 문장에 대한 음성 데이터를 송신해야 한다.
* 송신하지 못할 경우 일정 기간에 거쳐 반복하며, 그 장애를 로그에 기록해야 한다.

3.2.1.2.3 번역 데이터 수신(FR-1.2.3)

* 시스템은 Translation API에 문장을 보내어 번역 데이터를 수신 받아야 한다.
* 받지 못할 경우 일정 기간에 거쳐 반복하며, 그 장애를 로그에 기록해야 한다.

3.2.1.2.4 번역 데이터 송신(FR-1.2.4)

* 시스템은 요청된 문장에 대한 번역 데이터를 송신해야 한다.
* 송신하지 못할 경우 일정 기간에 거쳐 반복하며, 그 장애를 로그에 기록해야 한다.

3.2.1.2.5 기사 데이터 송신(FR-1.2.5)

* 시스템은 사용자의 요청된 기사에 대해 반환해야 한다.
* 송신하지 못할 경우 일정 기간에 거쳐 반복하며, 그 장애를 로그에 기록해야 한다.

3.2.1.3 User Management

3.2.1.3.1 로그인 요청 처리(FR-1.3.1)

* 시스템은 사용자의 로그인 요청에 대한 성공 실패 여부를 반환해야 한다.

3.2.1.3.2 로그아웃 처리(FR-1.3.2)

* 시스템은 사용자의 로그아웃 요청에 대한 성공 실패 여부를 반환해야 한다.

3.2.1.3.3 회원가입 요청 처리(FR-1.3.3)

* 시스템은 사용자의 회원 가입 요청에 대한 성공 실패 여부를 반환해야 한다.
* 시스템은 사용자의 회원 가입 정보를 자신의 DB에 저장해야 한다.

3.2.1.3.4 회원탈퇴 요청 처리(FR-1.3.4)

* 시스템은 사용자의 회원 탈퇴 요청에 대한 성공 실패 여부를 반환해야 한다.
* 시스템은 사용자의 회원 정보를 자신의 DB에서 삭제해야 한다.

3.2.1.3.5 회원 정보 수정 처리(FR-1.3.5)

* 시스템은 사용자의 회원 정보 수정 요청에 대한 성공 실패 여부를 반환해야 한다.
* 시스템은 사용자의 회원 정보를 수신된 정보로 수정해야 한다.

3.2.1.3.6 회원 정보 전송(FR-1.3.6)

* 시스템은 사용자의 회원 정보 요청에 대한 데이터를 반환해야 한다.

3.2.2 Frontend

3.2.2.1 Initialization

3.2.2.1.1 앱 실행(FR-2.1.1)

* 시스템은 실행 시 화면에 UI를 송출해야 한다.
* 유저의 종료 전까지 대기 상태를 유지해야 한다.

3.2.2.1.2 연결 유효성 검사(FR-2.1.2)

* 시스템은 실행 시 API Server와 통신이 원활한지 검사해야 한다.
* 원활하지 않을 경우 일정 기간에 거쳐 재시도 하며, 그 이력을 로그로 남겨야 한다.

3.2.2.2 Management

3.2.2.2.1 기사 제공(FR-2.2.1)

* 앱은 사용자가 선택한 기사를 제공해야 한다.
* 제공하지 못할 경우 대체 이미지를 제공한다.

3.2.2.2.2 마이페이지 접속(FR-2.2.2)

* 앱은 사용자의 정보를 제공할 수 있어야 한다.
* 제공하지 못할 경우 대체 이미지를 제공한다.

3.2.2.2.3 신문사 탭 전환(FR-2.2.3)

* 앱은 송출되는 기사 유형을 변경 가능해야 한다.

3.2.2.2.4 번역(FR-2.2.4)

* 앱은 사용자가 선택한 언어로 기사를 번역해야 한다.

3.2.2.2.5 음성 파일 제공(FR-2.2.5)

* 앱은 사용자가 선택한 기사에 대한 음성 파일을 제공해야 한다.

3.2.2.2.6 검색(FR-2.2.6)

* 앱은 사용자가 입력한 기사 제목으로 검색이 가능해야 한다.

3.2.2.2.7 즐겨찾기(FR-2.2.7)

* 앱은 사용자가 선택한 기사를 즐겨찾기 하여 모아볼 수 있게 해야 한다.

3.2.2.3 Auth

3.2.2.3.1 로그인(FR-2.3.1)

* 사용자는 자신이 입력한 정보를 통해 로그인이 가능해야 한다.
* 앱은 로그인의 성공 실패여부를 전송해야 한다.

3.2.2.3.2 로그아웃(FR-2.3.2)

* 사용자는 앱에 대해서 로그아웃이 가능해야 한다.
* 로그아웃 시 로그인 페이지로 넘어가야 한다.

3.2.2.3.3 회원가입(FR-2.3.3)

* 앱은 사용자의 정보를 받아 이를 서버에 등록할 수 있어야 한다.

3.2.2.3.4 회원탈퇴(FR-2.3.4)

* 앱은 사용자의 정보를 서버에서 제거할 수 있어야 한다.

3.2.3.3 Error

3.2.3.3.1 예외 처리(FR-3.3.1)

* 모든 예외 발생 시 일정 기간동안 반복해야 하며, 실패 시 사용자에게 Fallback 메시지를 제공해야 한다.

3.3 Performance Requirements

- 모든 로그는 전송 실패 시 최대 3회 재전송해야 한다.

3.4 Design Constraint

- 시스템은 Cloud Native로 개발되어야 하며, 각 플랫폼은 독립적이어야 한다.

- 로그는 SQL내에 저장되어야 한다.

- 시스템은 별다른 추가 조작 없이도 간편하게 사용자가 실행할 수 있어야 한다.

3.5 Software System Attributes

3.5.1 Reliability

- 예외 발생 시에도 시스템의 복구가 가능해야 한다.

- 각 모듈은 독립적인 예외 처리 로직을 구현해야 한다.

3.5.2 Availability

- API Server와 앱은 항상 동작가능한 상태를 유지해야 한다.

3.5.3 Security

- 모든 API 통신은 HTTPS프로토콜을 통해 암호화된다.

- 사용자 정보는 JWT, RT 기반의 인증을 수행한다.

3.5.4 Maintainability

- API Server와 Frontend는 모듈화된 구조로 개발되어 개발 서비스 수정 시 전체 시스템에 영향을 주지 않는다.

- 로그를 수집하여 운영 중 버그에 대한 수정이 가능해야 한다.

3.5.5 Portability

- API URL은 데이터 파일로 제공되어 이식성을 유지해야 한다.

- 웹 앱은 표준 브라우저에서 접속이 가능해야 한다.

- 모바일 앱은 SDK 35이상에서 접속이 가능해야 한다.