목차

[1. Preface 6](#_Toc513140778)

[1.1. Objective 6](#_Toc513140779)

[1.2. Readership 6](#_Toc513140780)

[1.2.1. User Requirement Readership 6](#_Toc513140781)

[1.2.2. System Requirement Readership 6](#_Toc513140782)

[1.3. Document Structure 6](#_Toc513140783)

[1.3.1. Preface 6](#_Toc513140784)

[1.3.2. Introduction 6](#_Toc513140785)

[1.3.3. Glossary 6](#_Toc513140786)

[1.3.4. User Requirement Definition 6](#_Toc513140787)

[1.3.5. System Architecture 7](#_Toc513140788)

[1.3.6. System Requirement Specification 7](#_Toc513140789)

[1.3.7. System Models 7](#_Toc513140790)

[1.3.8. System Evolution 7](#_Toc513140791)

[1.3.9. Appendices 7](#_Toc513140792)

[1.3.10. Index 7](#_Toc513140793)

[1.4. Version of the Document 7](#_Toc513140794)

[1.4.1. Version Format 7](#_Toc513140795)

[1.4.2. Version Management Policy 7](#_Toc513140796)

[1.4.3. Version Update History 7](#_Toc513140797)

[2. Introduction 8](#_Toc513140798)

[2.1. Objective 8](#_Toc513140799)

[2.2. Needs 8](#_Toc513140800)

[2.3. Our Game 8](#_Toc513140801)

[2.4. Expected Effect of the Service 10](#_Toc513140802)

[3. Glossary 10](#_Toc513140803)

[3.1. Objective 11](#_Toc513140804)

[3.2. Term Definitions, Acronyms and Abbreviations 11](#_Toc513140805)

[3.2.1. Term Definition 11](#_Toc513140806)

[A. User & Our game 관련 용어 11](#_Toc513140807)

[B. Development & Method 관련 용어 11](#_Toc513140808)

[C. 기타 관련 용어 11](#_Toc513140809)

[3.2.2. Acronym and Abbreviation 11](#_Toc513140810)

[4. User Requirement Definition 12](#_Toc513140811)

[4.1. Objective 12](#_Toc513140812)

[4.2. Functional Requirements 12](#_Toc513140813)

[4.2.1. 게임의 사회자 12](#_Toc513140814)

[A. Start / End Game 12](#_Toc513140815)

[B. Q & A Management 12](#_Toc513140816)

[C. Elimination 12](#_Toc513140817)

[D. Help / Log 13](#_Toc513140818)

[E. 전적 기록 13](#_Toc513140819)

[4.3. Non-Functional Requirements 13](#_Toc513140820)

[4.3.1. Product Requirement 13](#_Toc513140821)

[A. Performance Requiement 13](#_Toc513140822)

[B. Security Requirement 13](#_Toc513140823)

[C. Efficiency Requirement 13](#_Toc513140824)

[D. Dependability Requirement 14](#_Toc513140825)

[E. Usability Requirement 14](#_Toc513140826)

[4.3.2. Organizational Requirement 14](#_Toc513140827)

[A. Environment Requirement 14](#_Toc513140828)

[B. Operational Requirement 14](#_Toc513140829)

[5. System Architecture 15](#_Toc513140830)

[5.1. Objective 15](#_Toc513140831)

[5.2. Start System 15](#_Toc513140832)

[5.3. End System 15](#_Toc513140833)

[5.4. Q & A System 15](#_Toc513140834)

[5.5. Elimination System 15](#_Toc513140835)

[5.6. Help / log System 15](#_Toc513140836)

[5.7. DB management System 15](#_Toc513140837)

[6. System Requirement Specification 15](#_Toc513140838)

[6.1. Objective 15](#_Toc513140839)

[6.1.1. 게임의 사회자 15](#_Toc513140840)

[A. Start / End Game 15](#_Toc513140841)

[B. Q & A Management 16](#_Toc513140842)

[C. Elimination 16](#_Toc513140843)

[D. Help / Log 16](#_Toc513140844)

[E. 전적 기록 16](#_Toc513140845)

[6.2. Non-Functional Requirements 17](#_Toc513140846)

[6.2.1. Product Requirement 17](#_Toc513140847)

[A. Performance Requiement 17](#_Toc513140848)

[B. Security Requirement 17](#_Toc513140849)

[C. Efficiency Requirement 17](#_Toc513140850)

[D. Dependability Requirement 17](#_Toc513140851)

[E. Usability Requirement 18](#_Toc513140852)

[6.3. Scenario 18](#_Toc513140853)

[6.3.1. Join Scenario 18](#_Toc513140854)

[6.3.2. Question & Answer Scenario 18](#_Toc513140855)

[6.3.3. Elimination Scenario 18](#_Toc513140856)

[6.3.4. Winning Scenario 18](#_Toc513140857)

[6.3.5. Managing User profile Scenario 19](#_Toc513140858)

[7. System Models 19](#_Toc513140859)

[7.1. Objective 19](#_Toc513140860)

[7.2. Context Models 19](#_Toc513140861)

[A. ... 19](#_Toc513140862)

[B. ... 19](#_Toc513140863)

[C. .... 19](#_Toc513140864)

[D. .... 19](#_Toc513140865)

[E. .... 19](#_Toc513140866)

[F. ..... 19](#_Toc513140867)

[7.3. Interaction Models 19](#_Toc513140868)

[A. ... 19](#_Toc513140869)

[B. ... 19](#_Toc513140870)

[C. .... 19](#_Toc513140871)

[D. .... 19](#_Toc513140872)

[E. .... 19](#_Toc513140873)

[F. ..... 19](#_Toc513140874)

[7.4. Structural Models 19](#_Toc513140875)

[A. ... 19](#_Toc513140876)

[B. ... 19](#_Toc513140877)

[C. .... 20](#_Toc513140878)

[D. .... 20](#_Toc513140879)

[E. .... 20](#_Toc513140880)

[F. ..... 20](#_Toc513140881)

[7.5. Behavioral Models 20](#_Toc513140882)

[A. ... 20](#_Toc513140883)

[B. ... 20](#_Toc513140884)

[C. .... 20](#_Toc513140885)

[D. .... 20](#_Toc513140886)

[E. .... 20](#_Toc513140887)

[F. ..... 20](#_Toc513140888)

[8. System Evolutions 20](#_Toc513140889)

[8.1. Objective 20](#_Toc513140890)

[8.2. Limitations and Assumption 20](#_Toc513140891)

[8.3. Evolutions of User Requirement 20](#_Toc513140892)

[8.3.1. Voice Recognition-Based System 20](#_Toc513140893)

[8.3.2. Rule Modification Sandbox 20](#_Toc513140894)

[8.3.3. New game mods 20](#_Toc513140895)

[9. Appendices 20](#_Toc513140896)

[9.1. Objective 20](#_Toc513140897)

[9.2. Database Requirements 20](#_Toc513140898)

[9.3. User-System Requirements 21](#_Toc513140899)

[9.4. Development Process 21](#_Toc513140900)

[10. Index 21](#_Toc513140901)

[10.1. Table Index 21](#_Toc513140902)

[10.2. Figure Index 21](#_Toc513140903)

[10.3. Diagram Index 21](#_Toc513140904)

[11. Reference 21](#_Toc513140905)

1. Preface
   1. Objective

Preface에서는 본 문서의 예상되는 독자들과 문서의 전반적인 구조, 그리고 각 부분의 역할에 대하여 제시한다. 또한 버전 관리 정책, 버전 변경 기록, 그리고 문서의 변경사항들과 그에 대한 근거들을 서술한다.

* 1. Readership

본 요구사항 명세서는 독자에 따라 크게 User Requirement와 System Requirement의 두 부분으로 구성되어 있다.

* + 1. User Requirement Readership

User Requirement는 사용자의 관점에서 요구사항을 간략히 명세한 것이다. 본 프로젝트에서는 Game Player의 관점에서 서술한다.

* + 1. System Requirement Readership

System Requirement는 본 프로젝트에서 개발해야 하는 시스템의 특정 기능이 어떻게 구현되어야 하는지를 자세하게 기술한 것이다. 본 프로젝트에서는 Software Developer와 Server 및 Database Manager 관점에서 서술한다.

* 1. Document Structure

이 문서는 총 10개의 부분으로 구성되어 있다. Preface, Introduction, Glossary, User Requirements Definition, System Architecture, System Requirements Specification, System Models, System Evolution, Appendices, Index로 구성된다. 각 장의 역할과 전반적인 서술은 다음과 같다.

* + 1. Preface

Preface에서는 본 문서의 예상되는 독자들과 문서의 전반적인 구조, 그리고 각 부분의 역할에 대하여 서술한다. 또한 버전 관리 정책, 버전 변경 기록, 그리고 문서들의 변경사항과 그에 대한 근거들을 서술한다.

* + 1. Introduction

Introduction에서는 시스템의 필요성과 해당 시스템이 어떠한 Needs를 반영하고 있는지 서술한다. 또한 시스템의 기능이 다른 시스템과 어떻게 상호작용하는지 간략하게 서술한다.

* + 1. Glossary

Glossary에서는 본 문서에 등장하는 기술적 용어들과 축약어들을 정의한다.

* + 1. User Requirement Definition

User Requirement Definition에서는 사용자의 요구사항을 User Level에서 서술한다. 크게 Functional Requirement와 Non-Functional Requirement로 나누어 서술한다.

* + 1. System Architecture

System Architecture에서는 각 Component들의 hierarchy에 대하여 Diagram등을 이용해 서술한다.

* + 1. System Requirement Specification

System Requirement Specification에서는 User Level에서 정의되었던 Functional/Non-Functional Requirement들을 Developer Level로 세분화하여 서술한다.

* + 1. System Models

System Models에서는 System Component, System, 그리고 System Environment 사이의 관계를 Figure 및 Diagram을 통해 서술한다.

* + 1. System Evolution

System Evolution에서는 시스템이 세우고 있는 주요한 가정들과 사용자의 Needs 변화, 혹은 추후 Update될 사항들에 대해서 서술한다.

* + 1. Appendices

Appendices에서는 본 프로젝트와 관련한 참고사항들을 서술한다.

* + 1. Index

Index에서는 본 문서에 사용된 Table, Diagram, Fiqure들의 Index를 서술한다.

* 1. Version of the Document
     1. Version Format

Version 번호는 major.minor[.maintenance]로 구성되며, 문서의 버전은 0.1부터 시작한다.

* + 1. Version Management Policy

본 문서를 수정할 때마다 버전을 업데이트 한다. 이미 완성한 파트를 변경할 때는 Minor number를 변경하며, 새로운 부분을 추가하거나 괄목한 만한 변화가 있을 경우 Major number를 변경한다.

* + 1. Version Update History

|  |  |  |
| --- | --- | --- |
| Version | Committer | Explanation |
| 0.1 | 김동현 | 문서 목차 작성 및 Preface 서술. |
| 1.0 | 이용철 | Introduction 부분 서술 및 세부항목으로 Our game 파트 추가. |
| 2.0 | 김장훈 | User Requirement 일부 서술 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

1. Introduction
   1. Objective

Introduction에서는 시스템의 필요성과 해당 시스템이 어떠한 Needs를 반영하고 있는지 서술한다. 또한 시스템의 기능이 다른 시스템과 어떻게 상호작용하는지 서술한다.

* 1. Needs

사람들이 모여 다같이 즐길 수 있는 놀이로 보드 게임들을 많이 한다. 또한, 최근 몇 년 사이에 보드 보드게임 카페의 수가 증가하고 있고, 카페를 가면 여러 사람들이 모여 다양한 게임을 하는 모습, 심지어 손님이 너무 많아 자리가 없어 다른 카페를 찾아 나가는 사람들을 볼 수 있다. 보드 게임 중에서 마피아류 보드 게임을 하는 것을 심심치 않게 볼 수 있는데, 예를 들어 '뱅!', '스파이폴', '한밤의 늑대인간' 등을 하는 것을 볼 수 있다. 앞의 설명에서 보듯이, 우리는 보드 게임 중에서 마피아류 게임에 집중을 하고 있는데, 그 이유는 이 게임은 특별한 도구 없이 각자의 역할에 충실하여 속임을 통해 몇 명이든 다 같이 참여할 수 있는 게임으로 보드게임 카페뿐만 아니라, MT, 혹은 온라인에서 모여서 하기도 한다.

우리가 이 게임을 하면서 여러 문제를 겪을 수 있는데, 그 중 역할만 주어질 뿐 정보의 획득은 추리를 통해 서로의 심리를 읽어 얻는 것으로 정보의 불균형으로 인해 자칫하면 일방적으로 끝날 수 있다. 또한, 따로 사회자가 없어 게임의 진행을 참여자들이 진행을 간섭함으로써, 방향을 유리하게 만들어 가는 경우도 생긴다.

우리는 이에 따른 대안으로 게임의 사회자로 chatBot을 참여시키는 방법이다. chatBot이 사회자 역할을 함으로써, 게임을 흥미진진하게 진행이 될 수 있도록 상황에 맞추어 한쪽에 유리한 정보의 쏠림 방지하기 위해 약간의 정보의 제공을 해주는 기능, 참여자의 질문에 대해 다른 참여자의 질문에 따른 참 / 거짓의 판별, 투표 등의 행위를 처리 등의 기능을 제공할 계획이다.

* 1. Our Game

여기서 우리는 기존의 마피아 게임을 수정하여 제안한다. 우리의 게임은 다음의 규칙이 있다.

* 구성 : 사회자(chatBot), TEAM-A(king & members), TEAM-B(king & members)
* 각 참여자에 대해 갖는 제약조건들 :
  + .....
  + .....
  + ....
* 규칙 :
  + 각 참여자는 역할에 충실하면서 자유롭게 추리를 한다. 참여자는 자신이 속한 팀의 구성원이 누구인지 알 수 없다. 그 중에 사회자에게 질의를 할 수 있는 기회를 얻은 참여자가 모두에게 혹은 지목 & 선언을 통해 질의를 할 수 있다.
  + 사회자는 받은 질의를 모두에 공개하고 이에 대상자(들)은 답변을 해야 한다.
  + 사회자는 참여자의 제약조건과 규칙에 비교 후 결과를 공개한다.
  + 일정 시간 후에 사회자가 혹은 참여자 중 누군가 '투표!' 등을 발언 시 투표 시스템을 시작한다.
  + 팀의 왕이 죽으면 게임 종료 & 살아남은 왕의 팀이 승리.

우리가 제안한 게임 시스템은 Discord 기반에서 실행 될 것이며, 다음의 기능은 3가지의 기능 제공할 계획이다.

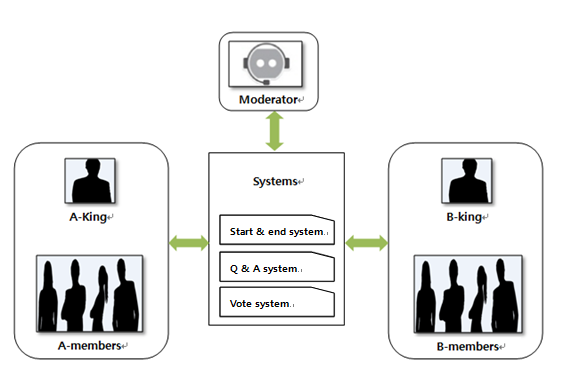
* 참여자의 역할 및 각 참여자에 대한 제약조건 랜덤 지정 기능 (Start System)
* 참여자의 질문 해석 및 다른 참여자들의 답변에 대한 참 & 거짓 판별 기능  
  (Q & A System)
* 투표 시스템 기능. (Vote System)

Figure1 Our game architecture

Start & End system : 게임의 시작과 엔드 조건을 충족하는지 관리하는 시스템이다. Start system은 참여자들에게 역할 & 제약 조건 & 시작은 위한 기본 초기화를 제공하기 위한 시스템이다. End System은 게임의 끝을 체크하기 위한 것으로, 왕이 죽었는지 체크하기 위한 것이다.

Q & A system : 참여자의 질의 및 답변을 관리하기 위한 시스템. 참여자의 질의를 해석 및 관리하고, 이를 다른 참여자들에게 통보. 이에 답변들을 관리하고, 해석을 통해 결과를 통보한다.

Vote system : 참여자들의 투표를 행하고 투표 결과를 통보하고, 많은 표를 받은 참여자를 게임에서 제외시키는 시스템이다.

* 1. Expected Effect of the Service
* Player
  + 정보의 불균형으로 인한 게임의 편향성을 최소화함으로써, 진행함에 있어서 흥미진진하게 게임을 즐길 수 있을 것으로 예상된다.

1. Glossary
   1. Objective

Glossary에서는 본 문서에 등장하는 기술적 용어들과 축약어들을 정의한다. 해당 문서를 어느 누가 읽어도 이해할 수 있도록 모든 용어에 대해 설명한다.

* 1. Term Definitions, Acronyms and Abbreviations
     1. Term Definition

3 가지의 분류로 Term Definitions을 서술한다.

|  |  |
| --- | --- |
| Terms | Definition |
| 참여자(Player) | 게임에 참여하는 사람들을 말한다. |
| 사회자(Moderator) | 게임의 진행을 제어하고 원활하게 진행을 이끄는 자를 말한다. |
| 왕(King) | 게임 내에서 한 팀의 왕으로, 생존을 함으로써 팀을 승리로 이끄는 자를 말한다. |
| 멤버(member) | 게임 내에서 한 팀의 왕을 제외한 구성원으로, 추리를 통해서 왕을 보호하고, 상대 구성원 및 왕을 죽이는 자를 말한다. |

* + - 1. User & Our game 관련 용어
      2. Development & Method 관련 용어

|  |  |
| --- | --- |
| Terms | Definition |
|  |  |
|  |  |
|  |  |
|  |  |

* + - 1. 기타 관련 용어

|  |  |
| --- | --- |
| Terms | Definition |
|  |  |
|  |  |
|  |  |
|  |  |

* + 1. Acronym and Abbreviation

|  |  |
| --- | --- |
| Terms | Definition |
|  |  |
|  |  |
|  |  |
|  |  |

1. User Requirement Definition
   1. Objective

User Requirement Definition에서는 사용자의 요구사항을 User Level에서 서술한다. 크게 Functional Requirement와 Non-Functional Requirement로 나누어 서술한다.

* 1. Functional Requirements

우리의 project에서 제공될 기능들은 다음과 같다.

* + 1. 게임의 사회자

[봇]은 사회자로서 다음의 기능들을 관리한다.

* + - 1. Start / End Game

Discord를 통해 게임을 시작하는 명령어가 입력될 경우, [봇]은 게임을 시작하기 위한 준비 과정에 들어간다.이 과정에서 게임에 참가하는 참여자,진행할 게임의 종류,세부 규칙 등의 설정을 할 수 있다.모든 준비가 끝났다면 준비완료 명령어를 입력하여 게임 시작단계로 들어간다.

게임이 시작되면 [봇]은 모든 참여자에게 팀과 역할 그리고 제약 조건을 부여한다.부여된 정보들은 참여자 본인만 확인할 수 있도록 개인 메시지를 통해 전달한다.

게임 도중 제외된 참여자의 역할이 왕이었다면,게임의 승패를 가릴 수 있게 된다. [봇]은 게임 종료를 선언하고, 결과를 발표한 뒤, 참여자들의 전적을 갱신한다.

* + - 1. Q & A Management

통상적인 게임 단계에서 [봇]은 참여자들의 문답을 관리해주는 상태가 된다.한 참여자가 Discord에 질문으로 여겨지는 text를 입력할 경우,[봇]은 질문을 해석한 뒤 질문 대상자에게 대답을 할 것을 요구한다.대답은 긍정 또는 부정 두 가지 형태가 존재할 수 있으며,그 외의 대답과 대상자가 아닌 참여자가 입력한 text들은 혼선을 피하기 위해 무시된다.질문 대상자가 대답 text를 입력하면, [봇]은 질문과 대답 그리고 대상자의 제약 조건을 비교하여 대상자가 올바른 대답을 하였는지 검증해준다.만약 질문 대상자가 부적절한 대답을 했을 경우 [봇]은 대상자에게 패널티를 부과한다.이 과정이 정상적으로 이루어진 뒤 [봇]은 다시 질문을 받아들이는 상태가 된다. 질문과 대답 text는 별도의 명령어 없이 자연어 형태로 입력할 수 있다.

* + - 1. Elimination

참여자가 다른 참여자를 지목하여 상대팀의 왕으로 지목하여 맞으면 승리로 이끌고, 틀리면, 본인의 퇴장.

* + - 1. Help / Log

Discord에 도움말 명령어를 입력하는 것으로 [봇]을 사용하기 위해 필요한 명령어나 게임의 룰을 볼 수 있다.또한,게임 도중에 로그 명령어를 입력하는 것으로 지금까지 이루어진 문답 내역을 볼 수 있다.

* + - 1. 전적 기록

참여자들의 승패 기록은 매 게임마다 자동적으로 데이터 베이스에 저장된다.프로필 명령어를 입력하는 것으로 자신의 게임 기록, 승률, 순위 등을 확인할 수 있으며, 원한다면 기록을 초기화할 수 있다. 전적 사항의 본인의 요청에 따라 공개 및 비공개로 설정할 수 있다.

* 1. Non-Functional Requirements
     1. Product Requirement
        1. Performance Requirement

Discord에 입력되는 text들을 빠르고 정확하게 처리하여 게임에 혼선을 빚는 일이 없어야 한다. 자연어 text를 해석해야 하는 경우, [봇]이 오역을 하는 일이 없도록 확실한 학습이 필요하다

* + - 1. Security Requirement

[봇]을 사용할 때 발생할 수 있는 개인정보와 보안 문제점들은 대부분 Discord의 시스템에 기인한다. [봇]과 사용자 사이에는 항상 Discord가 존재하기 때문에, Discord 상에 보안 문제가 없다면 [봇]을 사용할 때도 문제가 발생하지 않는다. [봇]이 자연어를 처리하는 과정에서 개인 정보가 포함된 text를 데이터 베이스에 저장하는 일을 방지하기 위해서, 모든 text는 게임에 연관된 정보만을 논리식으로 번역하여 다루도록 한다.

[수정이 필요할 것 같음.]

* + - 1. Efficiency Requirement

[봇]은 서버에 의해서 항시 실행되는 상태를 유지하고 있다. 따라서, 서버에 부하가 걸려 [봇]의 작동 효율이 나빠지는 것을 방지하기 위해 이동하는 데이터의 양을 최소화해야 한다. 복잡한 연산이 필요한 부분은 [봇]내부에서 처리하는 것 보다 외부 API를 적극적으로 활용하여 서버 유지에 필요한 자원을 줄인다..

* + - 1. Dependability Requirement

사람이 사회자를 맡는 경우와 다르게 bot은 게임 도중 발생한 문제를 적절히 해결하고 조율하는 능력이 없다. 따라서, [봇]이 조금이라도 잘못 작동하면 게임의 전체적인 양상이 이상해질 수 있다. 전체적인 게임의 진행을 bot에게 맡기는 이상 [봇]의 dependability가 확실해야 한다.

* + - 1. Usability Requirement

별도의 가이드라인을 제공하지 않아도, [봇]을 사용하는 데 어려움이 없어야 한다. [봇]이 Discord 상에 메시지를 출력할 때마다, 도움말 명령어를 입력하여 자세한 정보를 볼 수 있다는 문구를 첨가하여, 사용자가 도움말을 읽고 [봇]을 사용하는 방법을 익힐 것을 권고한다. 게임 도중 진행 상황을 참여자들이 혼동하지 않도록, 현황 정보를 알려주는 메시지를 눈에 띄게 출력한다.

최대한 명령어의 의존하지 않고, 즉 자연어로서의 명령을 잘 처리하도록 개발해야 한다.

* + 1. Organizational Requirement
       1. Environment Requirement

Discord bot으로서 [봇]의 대부분 기능은 Discord API에 묶여있다.따라서, [봇]은 Discord에서 사용하기에 최적화되어 있어야 한다. Discord는 PC 프로그램과 모바일 어플리케이션 두 종류의 플랫폼을 지원한다. [봇]은 두 플랫폼 모두에서 사용될 것을 고려하여 만들어져야 한다. 즉, 가독성 초점을 두고 모바일 환경과 데스트탑 환경에서의 가독성을 동등한 수준으로 유지해야 한다.

* + - 1. Operational Requirement

[봇]은 자연어의 해석을 위해 Dialogflow API 버전의 업데이트와 자연스러운 자연어 해석을 위해 지속적인 학습을 시킬 수 있도록 설계한다.

1. System Architecture
   1. Objective

System Architecture에서는 각 Component들의 hierarchy에 대하여 Diagram등을 이용해 서술한다.

* 1. Start System
  2. End System
  3. Q & A System
  4. Elimination System
  5. Help / log System
  6. DB management System

1. System Requirement Specification
   1. Objective

System Requirement Specification에서는 User Level에서 정의되었던 Functional/Non-Functional Requirement들을 Developer Level로 세분화하여 서술한다.

* + 1. 게임의 사회자

[봇]은 사회자로서 다음의 기능들을 관리한다.

* + - 1. Start / End Game

*// user requirement*

*Discord를 통해 게임을 시작하는 명령어가 입력될 경우, [봇]은 게임을 시작하기 위한 준비 과정에들어간다.이 과정에서 게임에 참가하는 참여자,진행할 게임의 종류,세부 규칙 등의 설정을 할 수 있다.모든 준비가 끝났다면 준비완료 명령어를 입력하여 게임 시작단계로 들어간다.*

*게임이 시작되면 [봇]은 모든 참여자에게 팀과 역할 그리고 제약 조건을 부여한다.부여된 정보들은 참여자 본인만 확인할 수 있도록 개인 메시지를 통해 전달한다.*

*게임 도중 제외된 참여자의 역할이 왕이었다면,게임의 승패를 가릴 수 있게 된다. [봇]은 게임 종료를 선언하고, 결과를 발표한 뒤, 참여자들의 전적을 갱신한다.*

* + - 1. Q & A Management

*//user re*

*통상적인 게임 단계에서 [봇]은 참여자들의 문답을 관리해주는 상태가 된다.한 참여자가 Discord에 질문으로 여겨지는 text를 입력할 경우,[봇]은 질문을 해석한 뒤 질문 대상자에게 대답을 할 것을 요구한다.대답은 긍정 또는 부정 두 가지 형태가 존재할 수 있으며,그 외의 대답과 대상자가 아닌 참여자가 입력한 text들은 혼선을 피하기 위해 무시된다.질문 대상자가 대답 text를 입력하면, [봇]은 질문과 대답 그리고 대상자의 제약 조건을 비교하여 대상자가 올바른 대답을 하였는지 검증해준다.만약 질문 대상자가 부적절한 대답을 했을 경우 [봇]은 대상자에게 패널티를 부과한다.이 과정이 정상적으로 이루어진 뒤 [봇]은 다시 질문을 받아들이는 상태가 된다. 질문과 대답 text는 별도의 명령어 없이 자연어 형태로 입력할 수 있다.*

* + - 1. Elimination

*// user re*

*참여자가 다른 참여자를 지목하여 상대팀의 왕으로 지목하여 맞으면 승리로 이끌고, 틀리면, 본인의 퇴장*.

* + - 1. Help / Log

*// user re.*

*Discord에 도움말 명령어를 입력하는 것으로 [봇]을 사용하기 위해 필요한 명령어나 게임의 룰을 볼 수 있다.또한,게임 도중에 로그 명령어를 입력하는 것으로 지금까지 이루어진 문답 내역을 볼 수 있다.*

* + - 1. 전적 기록

// *user Re.*

*참여자들의 승패 기록은 매 게임마다 자동적으로 데이터 베이스에 저장된다.프로필 명령어를 입력하는 것으로 자신의 게임 기록, 승률, 순위 등을 확인할 수 있으며, 원한다면 기록을 초기화할 수 있다. 전적 사항의 본인의 요청에 따라 공개 및 비공개로 설정할 수 있다.*

* 1. Non-Functional Requirements
     1. Product Requirement
        1. Performance Requiement

Discord에 입력되는 text들을 빠르고 정확하게 처리하여 게임에 혼선을 빚는 일이 없어야 한다. 자연어 text를 해석해야 하는 경우, [봇]이 오역을 하는 일이 없도록 확실한 학습이 필요하다

예 ) 1초 이내에 각 프로세싱에 대해 결과를 출력할 수 있도록..

* + - 1. Security Requirement

[봇]을 사용할 때 발생할 수 있는 개인정보와 보안 문제점들은 대부분 Discord의 시스템에 기인한다. [봇]과 사용자 사이에는 항상 Discord가 존재하기 때문에, Discord 상에 보안 문제가 없다면 [봇]을 사용할 때도 문제가 발생하지 않는다. [봇]이 자연어를 처리하는 과정에서 개인 정보가 포함된 text를 데이터 베이스에 저장하는 일을 방지하기 위해서, 모든 text는 게임에 연관된 정보만을 논리식으로 번역하여 다루도록 한다.

[수정이 필요할 것 같음.]

* + - 1. Efficiency Requirement

[봇]은 서버에 의해서 항시 실행되는 상태를 유지하고 있다. 따라서, 서버에 부하가 걸려 [봇]의 작동 효율이 나빠지는 것을 방지하기 위해 이동하는 데이터의 양을 최소화해야 한다. 복잡한 연산이 필요한 부분은 [봇]내부에서 처리하는 것보다 외부 API를 적극적으로 활용하여 서버 유지에 필요한 자원을 줄인다.

communication에 필요한 data structure의 구조의 attr들을 최대 5~6개로 함으로써 json 객체의 크기를 줄여 서버와 클라이언트 간의 통신을 효율적으로 한다.

* + - 1. Dependability Requirement

*// 아래의 문단을 삭제하거나 수정이 필요.*

*[봇]에게 충분한 자연어 해석 학습을 시켜도 오역을 하는 경우가 발생할 수 있다. 이런 경우 봇이 해석한 바를 참여자에게 전달하지 않으면, 사용자의 해석과 시스템의 해석 사이에서 발생한 차이가 메워지지 않은 채로 게임을 혼란스럽게 만들 수 있다. 이를 방지하기 위해서 [봇]은 해석한 바를 참여자가 볼 수 있게 출력한다. 오역이 발생하지 않은 경우 [봇]이 올바르게 작동하고 있음을 시각적으로 확인할 수 있고, 오역이 발생하더라도 참여자가 정정할 기회를 줘서 [봇]에 대한 신뢰도가 높아진다.*

사람이 사회자를 맡는 경우와 다르게 bot은 게임 도중 발생한 문제를 적절히 해결하고 조율하는 능력이 없다. 따라서, [봇]이 조금이라도 잘못 작동하면 게임의 전체적인 양상이 이상해질 수 있다. 전체적인 게임의 진행을 bot에게 맡기는 이상 [봇]의 dependability가 확실해야 한다.

자연어에서 논리식으로 해석 및 번역의 정확도를 90% 이상으로 해야 한다.

* + - 1. Usability Requirement

별도의 가이드라인을 제공하지 않아도, [봇]을 사용하는 데 어려움이 없어야 한다. [봇]이 Discord 상에 메시지를 출력할 때마다, 도움말 명령어를 입력하여 자세한 정보를 볼 수 있다는 문구를 첨가하여, 사용자가 도움말을 읽고 [봇]을 사용하는 방법을 익힐 것을 권고한다. 게임 도중 진행 상황을 참여자들이 혼동하지 않도록, 현황 정보를 알려주는 메시지를 눈에 띄게 출력한다.

명령어의 종류의 사용법, 즉 도움 명령어의 사용법, 전적 확인 및 보기 명령어의 사용법 등을 보여준다. 예) !help , !request

* 1. Scenario
     1. Join Scenario

사용자가 여러 명 모인 챗방에서 [봇]을 초대하면, [봇]이 '무슨 게임을 시작 하시겠습니까' 로 하여금 게임 종류를 받고, 참여 인원 조사를 하고 게임을 시작하는 메시지를 출력하고, 이에 따라 [봇]이 참여자의 역할 배정 등의 초기화 작업을 한다.

* + 1. Question & Answer Scenario

참여자가 한 명이 전체에게 질문을 하던지 혹은 한 명을 지목하여 질문을 할 수 있다. 이에 지목 당한 사람(들)은 제약 조건에 따라서 답변을 하고, 이 답변(들)을 [봇]이 이 부분에서 대해서 제재를 해야 한다.

* + 1. Elimination Scenario

어느 한 참여자가 다른 참여자 한 명을 지목하여 '너 상대방 팀 왕이지!'라고 하면, 봇이 이것을 확인을 하고 지목 당한 참여자가 왕이 아닐 경우, 지목한 참여자를 제거(제외)시킨다 혹은 왕이 맞을 경우 지목한 참여자가 있는 팀이 승리한다.

* + 1. Winning Scenario

모든 참여자의 전적 사항을 업데이트를 해서 DB 안에 데이터의 갱신이 이뤄져야하고, [봇]은 승리 팀의 승리를 했다는 것을 메시지로 출력한다. 사용자가 원할 경우 새로운 게임을 시작하고 처음 과정으로 되돌아간다.

* + 1. Managing User profile Scenario

User가 본인의 전적 사항을 요청 시에 DB에 저장된 그 user의 정보를 가져와 출력한다. 초기화를 원할 경우 초기화 해준다.

*/// 상대의 전적 사항을 보여주는 건 보류*

1. System Models
   1. Objective

System Models에서는 System Component, System, 그리고 System Environment 사이의 관계를 Figure 및 Diagram을 통해 서술한다.

* 1. Context Models
     + 1. ...
       2. ...
       3. ....
       4. ....
       5. ....
       6. .....
  2. Interaction Models
     + 1. ...
       2. ...
       3. ....
       4. ....
       5. ....
       6. .....
  3. Structural Models
     + 1. ...
       2. ...
       3. ....
       4. ....
       5. ....
       6. .....
  4. Behavioral Models
     + 1. ...
       2. ...
       3. ....
       4. ....
       5. ....
       6. .....

1. System Evolutions
   1. Objective

System Evolution에서는 시스템이 세우고 있는 주요한 가정들과 사용자의 Needs 변화, 혹은 추후 Update될 사항들에 대해서 서술한다.

* 1. Limitations and Assumption
  2. Evolutions of User Requirement
     1. Voice Recognition-Based System
     2. Rule Modification Sandbox
     3. New game mods

1. Appendices
   1. Objective

Appendices에서는 본 프로젝트와 관련한 참고사항들을 서술한다.

* 1. Database Requirements
  2. User-System Requirements
  3. Development Process

1. Index
   1. Table Index
   2. Figure Index
   3. Diagram Index
2. Reference