프로젝트 계획서

<Eight Eureka 8조>

- 변 경 이 력 -

|  |  |  |  |
| --- | --- | --- | --- |
| 일자 | 버전 | 변경 내역 | 작 성 자 |
| 17.05.01 | V0.01 | 개요 및 WBS 초안 작성. | 김정원 |
| 17.05.04 | V0.02 | 산출물 선정  역할 선정  PERT차트 초안 | 강성연  김상규  김정원  민병석  최수빈 |
| 17.05.05 | V0.03 | Gantt 차트 초안  일정 세분화 추가  산출물 관리 작성 | 김정원 |
| 17.05.09 | V0.04 | WBS 수정  Gantt , PERT 차트 수정  조직도, 역할 및 책임 수정  도구, 일정 수정 | 김정원 |
| 17.05.10 | V1.0 | 위험 관리 계획 작성  회의를 통한 버전 동결. | 김정원 |
|  |  |  |  |
|  |  |  |  |

- 목 차 -

[1. 프로젝트 개요 - 3 -](#_Toc447537505)

[1.1 목적 - 3 -](#_Toc447537506)

[1.2 주요 일정 - 3 -](#_Toc447537507)

[1.3 조직 - 4 -](#_Toc447537508)

[1.3.1 조직도 - 4 -](#_Toc447537509)

[1.3.2 역할 및 책임 - 4 -](#_Toc447537510)

[1.4 생명주기 모델 - 5 -](#_Toc447537511)

[1.5 도구 - 5 -](#_Toc447537512)

[2. 규모 산정 - 6 -](#_Toc447537513)

[2.1 WBS(Work Breakdown Structure) - 6 -](#_Toc447537514)

[3. 일정 - 7 -](#_Toc447537515)

[4. 산출물 관리 - 8 -](#_Toc447537516)

[5. 위험 관리 계획 - 9 -](#_Toc447537517)

# 프로젝트 개요

## 목적

|  |
| --- |
| 프로젝트에 대한 간단한 소개 및 수행 목적을 기술한다. |

본 프로젝트의 목적은 대학생을 위한 과목별 To do list 관리 프로그램 제작이다.

본 프로젝트 계획서는 개발 단계에 대한 일정 산출과 효율적인 개발 능률을 위한 조직도 구성 및 역할 분담과 생명주기 표준, 사용 도구 표준을 기술하는 프로젝트의 중심 문서이다.

## 주요 일정

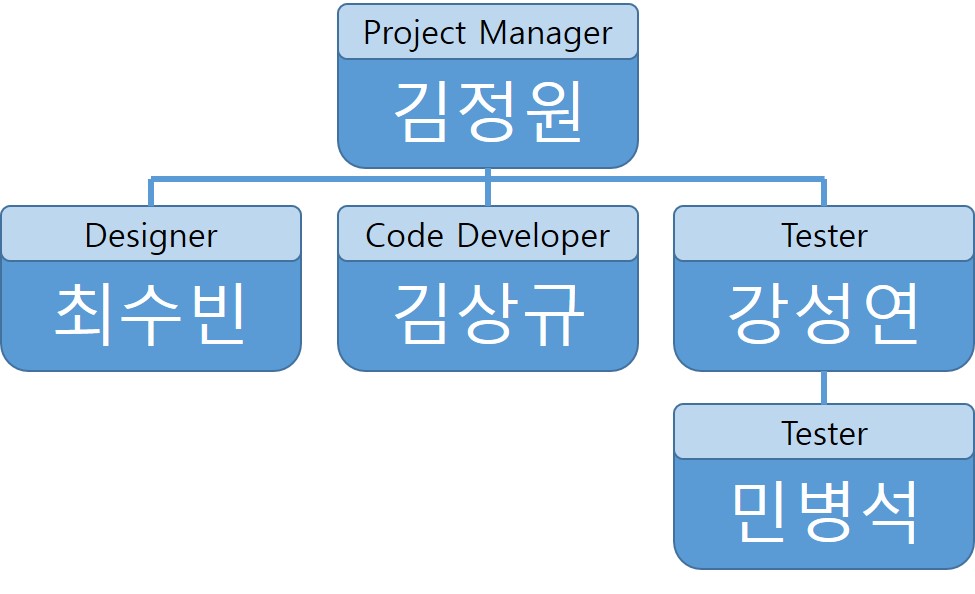
|  |
| --- |
| 프로젝트에서 주어진 주요 단계, 일정 별 산출물을 기술한다. |

|  |  |  |
| --- | --- | --- |
| 단계 | 일정 | 산출물 |
| 설계 | 17.05.22(월) | 설계서.doc |
| 구현 | 17.06.02(금)(가변) | 프로그램 소스코드 |
| 평가 | 17.06.07(수) | 테스트 케이스  & 테스트 결과 보고서 |
| 시연 | 17.06.12(월) | 최종 발표.ppt |

## 조직

|  |
| --- |
| 프로젝트 내 구성원들의 조직도 및 역할을 기술한다. |

### 조직도



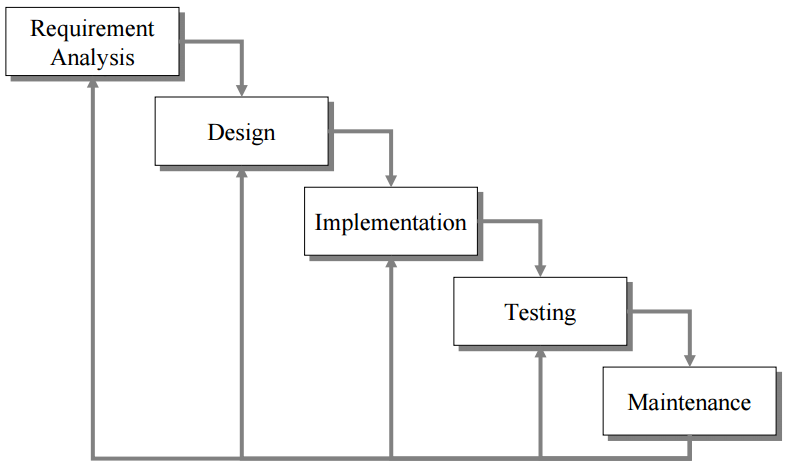
### 역할 및 책임

|  |  |  |
| --- | --- | --- |
| 팀원 | 역할 | 책임 |
| 강성연 | Tester | 코드 제작자의 결과물에 대해 정확한 테스트와 피드백을 제공한다.  오류 검출 및 수정 사항에 대한 문서를 책임 한다. |
| 김상규 | Code Developer | 요구사항을 기반한 설계서를 만족시키는 기능을 수행 하는 소스코드 제작을 책임 한다. |
| 김정원 | PM | 프로젝트 전반적인 진행 경과를 모니터링하고 컨트롤하며 일감에 대한 분배를 책임 하고, 구성원들이 능력을 잘 발휘 할 수 있도록 준비한다. |
| 민병석 | Tester | 코드 제작자의 결과물에 대해 정확한 테스트와 피드백을 제공한다.  오류 검출 및 수정 사항에 대한 문서를 책임 한다. |
| 최수빈 | Designer | 요구사항에 만족하며 코드 제작자가 이해할 수 있는 정확한 설계서 작성을 책임 한다. |

## 생명주기 모델

|  |
| --- |
| 프로젝트에 적용할 생명주기에 대한 설명을 기술한다. |

본 프로젝트에서의 생명주기는 Waterfall 방식을 취한다.



완료된 요구사항 명세서를 바탕으로 디자인을 완성하고 완성된 디자인을 바탕으로 개발에 착수한다. 이어서 테스트에 대한 충분한 피드백을 거쳐 프로젝트를 진행한다.

## 도구

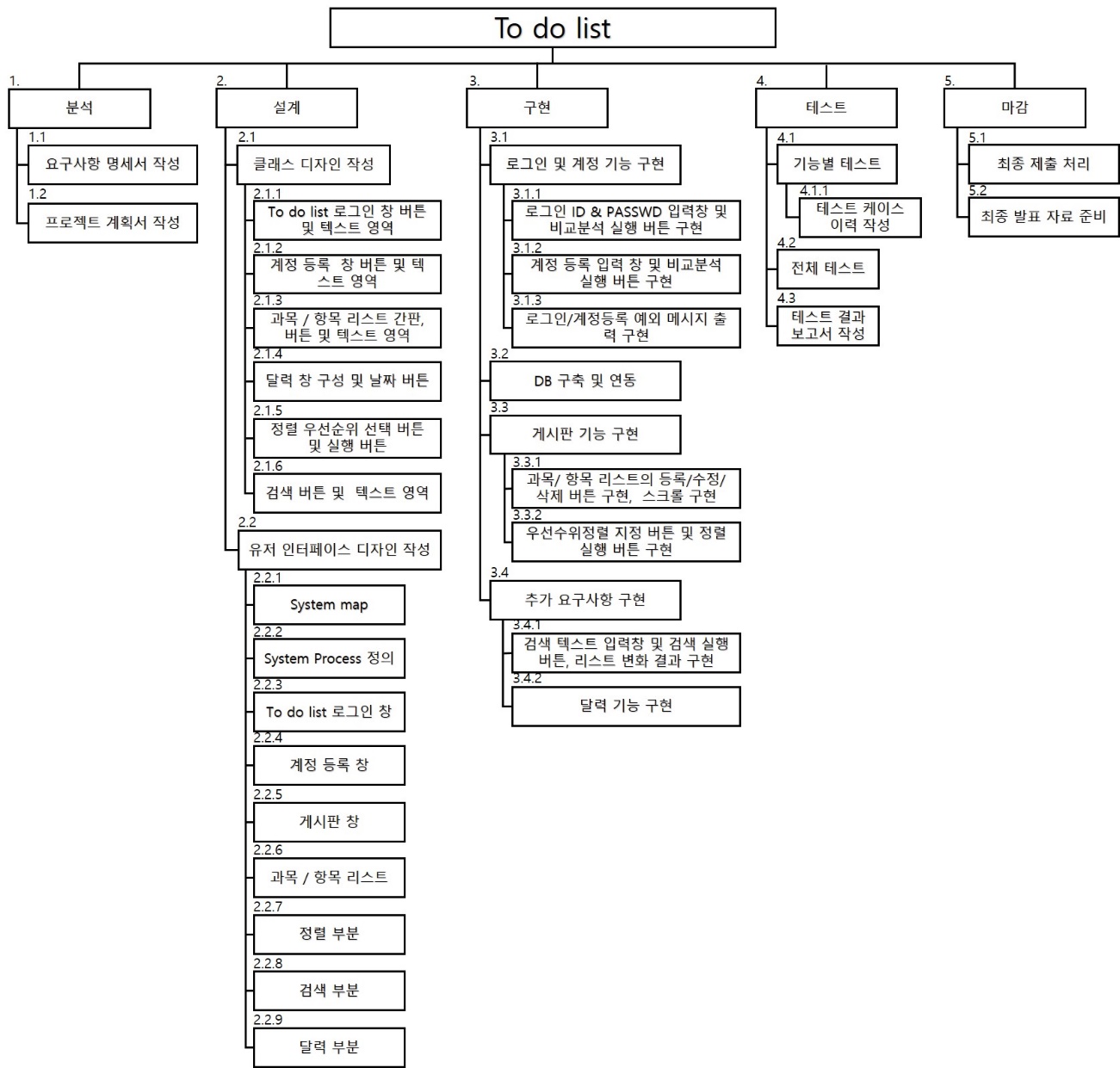
|  |
| --- |
| 프로젝트에 적용할 도구에 대한 설명을 기술한다.  Ex) 일정관리 도구, 형상관리 도구 등.. |

|  |  |  |
| --- | --- | --- |
| 일정 관리 | Redmine | Usecase 일감에 대한 분배와 진도 추적, 모니터링에 활용한다. |
| 형상 관리 | Tortoise Git | 모든 문서의 버전 관리와 작업자들 간의 공유에 활용한다. |
| 코드 작성 | Eclipse (Java) | 소스코드 작성 간에 프로그래밍 언어와 도구의 표준을 정한다. |
| 문서 도구 | MS Word,  MS PowerPoint,  MS Excel | 모든 문서 도구와 양식을 MS Word로 통일 한다.  각종 다이어그램과 차트는 MS PowerPoinot와 MS Excel을 활용한다. |

# 규모 산정

|  |
| --- |
| WBS를 작성하고, 각 작업에 소요되는 기간을 계산한다. |

## WBS(Work Breakdown Structure)

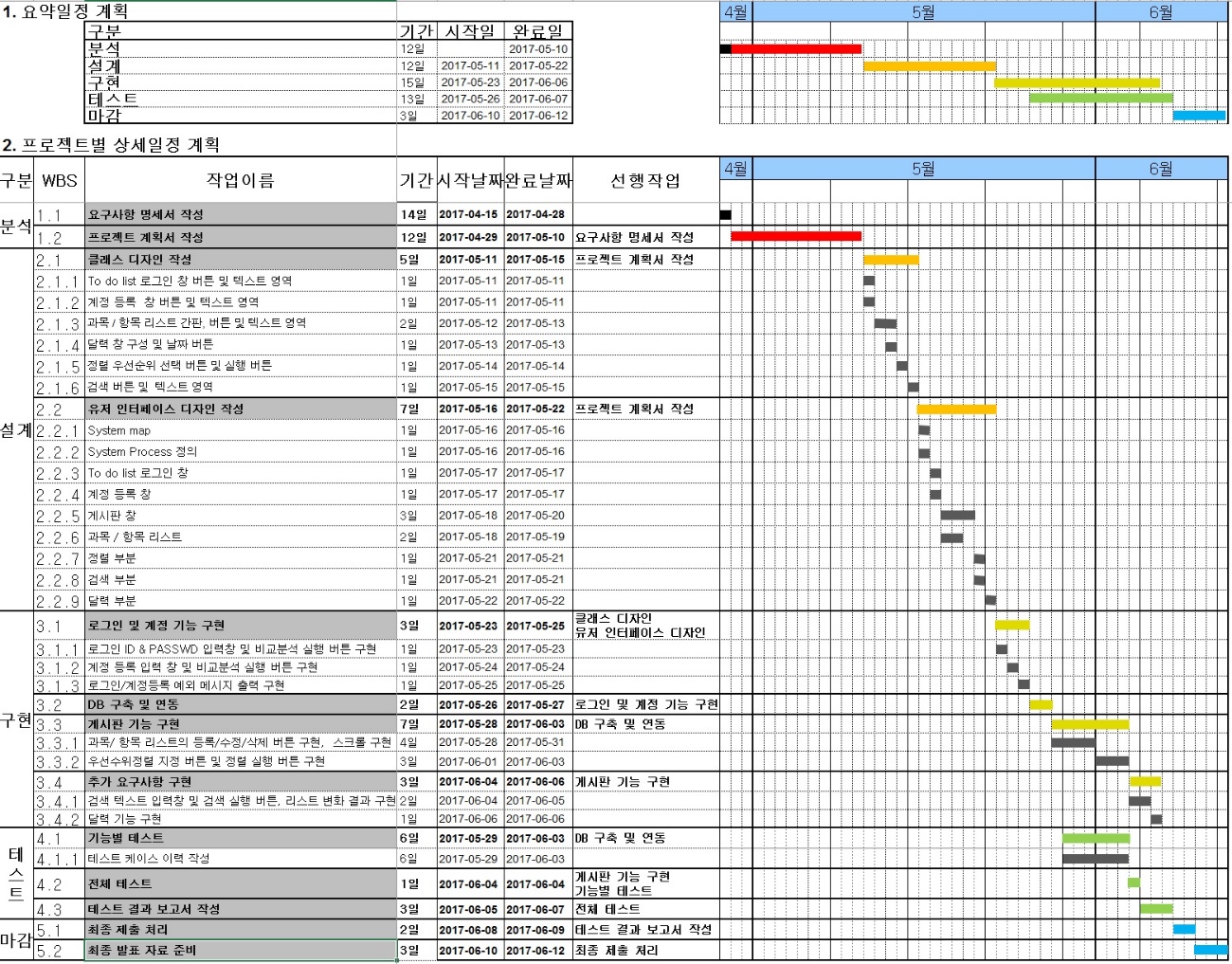


# 일정

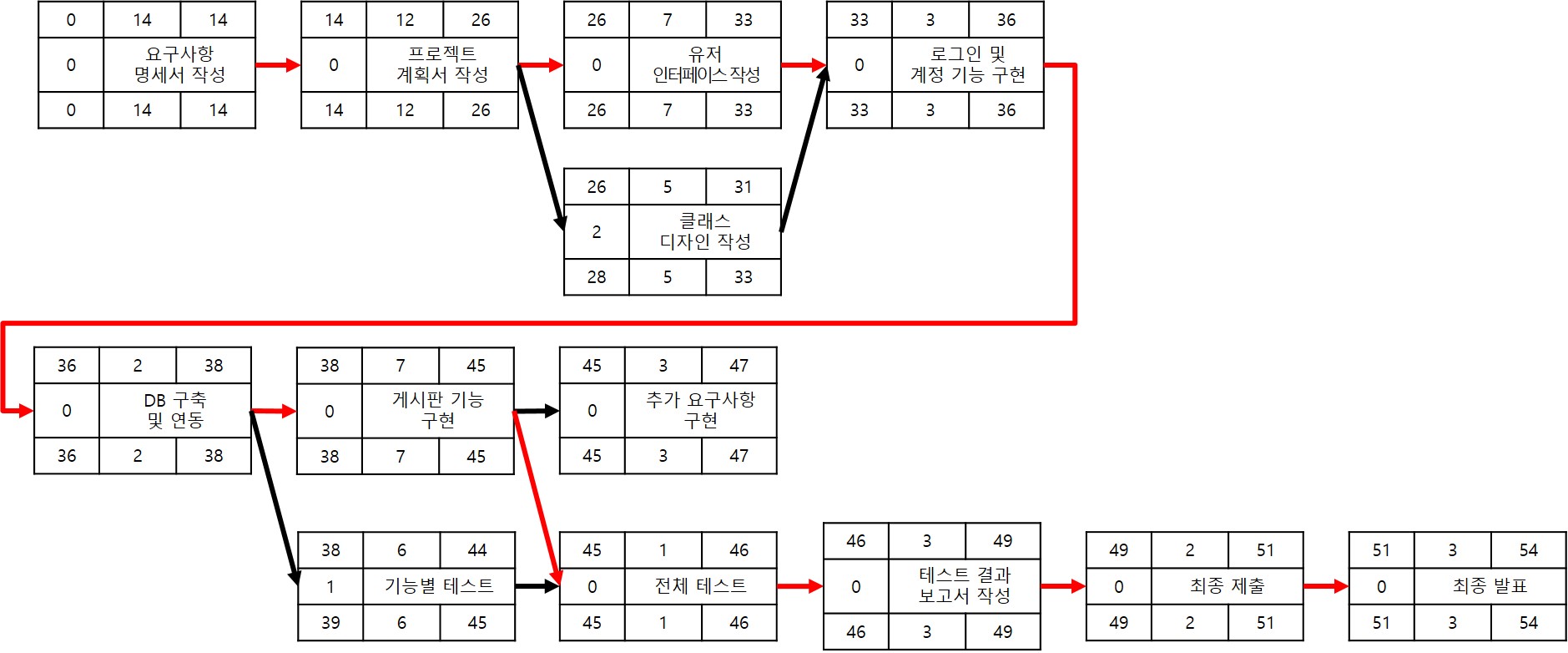
|  |
| --- |
| 주요 일정을 Gantt 차트와 PERT 차트를 이용하여 작성한다. |

|  |  |  |
| --- | --- | --- |
| 작업 명 | 소요시간 | 선행작업 |
| 요구사항 명세서 작성 | 14일 (4/15~4/28) | - |
| 프로젝트 계획서 작성 | 12일 (4/29 ~ 5/10) | 요구사항 명세서 작성 |
| 클래스 디자인 | 5일 (5/11 ~ 5/15) | 프로젝트 계획 |
| 유저 인터페이스 디자인 | 7일 (5/16 ~ 5/22) | 프로젝트 계획 |
| 로그인 및 계정 기능 구현 | 3일(5/23 ~ 5/25) | 클래스 디자인  유저 인터페이스 디자인 |
| DB 구축 및 연동 | 2일(5/26 ~ 5/27) | 로그인 및 계정 기능 구현 |
| 게시판 기능 구현 | 7일(5/28 ~ 6/3) | DB 구축 및 연동 |
| 추가 요구사항 구현 | 3일(6/4 ~ 6/6) | 게시판 기능 구현 |
| 기능별 테스트 | 6일(5/29 ~ 6/3) | DB 구축 및 연동 |
| 전체 테스트 | 1일(6/4) | 게시판 기능 구현  기능별 테스트 |
| 테스트 결과 보고서 작성 | 3일(6/5 ~ 6/7) | 전체 테스트 |
| 최종 제출 | 2일(6/8 ~ 6/9) | 테스트 결과 보고서 |
| 최종 발표 | 3일(6/10 ~ 6/12) | 최종 제출 |

## Gantt 차트



## PERT 차트



# 산출물 관리

|  |
| --- |
| 산출물의 관리 방안을 기술한다.  산출물을 어디에 저장하고, 산출물 명을 어떤 규칙으로 할지 등을 기술한다. |

모든 산출물은 Git 저장소에 공유하여 모든 작업자들이 열람하거나 수정할 수 있게 한다.

또한 Redmine 서버를 통해 일감을 관리하여 산출물이 계획적이고 체계적 완성 될 수 있게 한다. 문서 산출물은 파일 제목 뒤와 문서 내 변경 이력과 머리말에도 버전을 동일하게 명시하여 문서 작성 간 Git으로 자동 버전 관리를 한다.

변경 이력은 문서 수정한 사람이 작성하며 정확하고 간결하게 명시한다.

문서 버전은 v0.01로 초안을 시작하고 부분적인 변경 사항은 소수 둘째 자리를 상승시킨다.

문서 완성도가 일정 수준에 올라 다음 작업을 위한 Baseline이 되면 v1.0으로 동결 후 추후 변경 수요가 생길 시 오직 회의를 통해서만 접근한다.

# 위험 관리 계획

|  |
| --- |
| 프로젝트 수행 중에 발생할 위험을 식별하고, 이를 대처하기 위한 방안을 기술한다. |

|  |  |  |  |
| --- | --- | --- | --- |
| 위험 요소 | 가능성 | 영향도 | 대처 방안 |
| 기간 내에 코딩을 완료하지 못하는 경우 | 30% | 상 | 모니터링 간 개발 진도의 누락이 예측 되면 전체 코딩 일감에 중 “추가 요구사항”을 포기하고 핵심 요구사항들에 대해 최대한 집중한다. |
| 문서작업, 소스코드 작업 간 데이터를 잃은 경우 | 5% | 하 | 형상관리 도구로 채택한 Tortoise Git으로 철저한 데이터 원본 및 버전 관리를 통해 손실에 대한 가능성을 최대한 줄이고 문서 작업 시에는 복사본을 통해 수정하여 검토 후 commit한다. |
| 구성원이 질병 및 사고로 더 이상 프로젝트 참여가 제한된 경우 | 3% | 중 | 해당 인원의 업무를 수행 가능한 복수의 대체 인원들을 선정하여 역할을 분담한다. |
| 개발 능력의 한계성으로 인해 요구사항을 충족하지 못할 경우 | 20% | 상 | 1안으로 외부의 월등한 수준의 개발 자문위원을 영입하여 컨설팅 받는다.  2안으로 구현 가능한 한계 수준까지 최대한 개발 하되 정확하게 구현하여 추후 미흡 부분에 대한 시간과 비용을 재협상한다. |
| 구성원 간 의사소통 및 정보공유의 미흡이나 과도한 문서화로 정보 전달이 어려움이 생기는 경우 | 60% | 중 | PM이 일의 문제점을 정확히 분석하여 주간회의에서 구성원들이 의견을 제대로 공유하고 나눌 수 있도록 회의 내용을 수준 높게 준비시킨다. |
| 개발 도중 요구사항이 변경되는 경우 | 90% | 최상 | 긴급 회의를 통해 요구사항 변화에 따라 적응하여 모든 문서와 소스코드를 추가 및 수정 한다.  이를 빠르고 순탄하게 적응시키려면 소스코드 개발 초기부터 추가 및 수정 요구사항에 대해 적응력이 높도록 객체지향적인 각도로 구현해야 한다. |