프로젝트 계획서

**<THREEGO>**

- 변 경 이 력 -

|  |  |  |  |
| --- | --- | --- | --- |
| 일자 | 버전 | 변경 내역 | 작 성 자 |
| 2018-03-30 | V1.0.1 | 퍼트 차트 수정 | 김정현 |
| 2018-04-12 | V1.0.2 | 설명을 위한 네모박스 삭제  폰트 크기 조절 | 김인환 |
| 2018-05-30 | V1.0.3 | 최종 수정 | 김정현 |
| 2018-05-31 | V1.0.4 | 간트 차트 해상도 수정 | 김정현 |
| 2018-05-31 | V1.0.5 | PERT 차트 해상도 수정 | 김정현 |
| 2018-05-31 | V1.0.6 | 버전 정보 수정 | 김정현 |
|  |  |  |  |

- 목 차 -

[1. 프로젝트 개요 - 3 -](#_Toc508731616)

[1.1 목적 - 3 -](#_Toc508731617)

[1.2 주요 일정 - 3 -](#_Toc508731618)

[1.3 조직 - 4 -](#_Toc508731619)

[1.3.1 조직도 - 4 -](#_Toc508731620)

[1.3.2 역할 및 책임 - 4 -](#_Toc508731621)

[1.4 생명주기 모델 - 5 -](#_Toc508731622)

[1.5 도구 - 5 -](#_Toc508731623)

[2. 규모 산정 - 6 -](#_Toc508731624)

[2.1 WBS(Work Breakdown Structure) - 6 -](#_Toc508731625)

[3. 일정 - 7 -](#_Toc508731626)

[4. 산출물 관리 - 8 -](#_Toc508731627)

[5. 리스크 관리 계획 - 9 -](#_Toc508731628)

# 프로젝트 개요

## 목적

대학생을 위한 과목별 TO DO LIST 관리 프로그램 개발

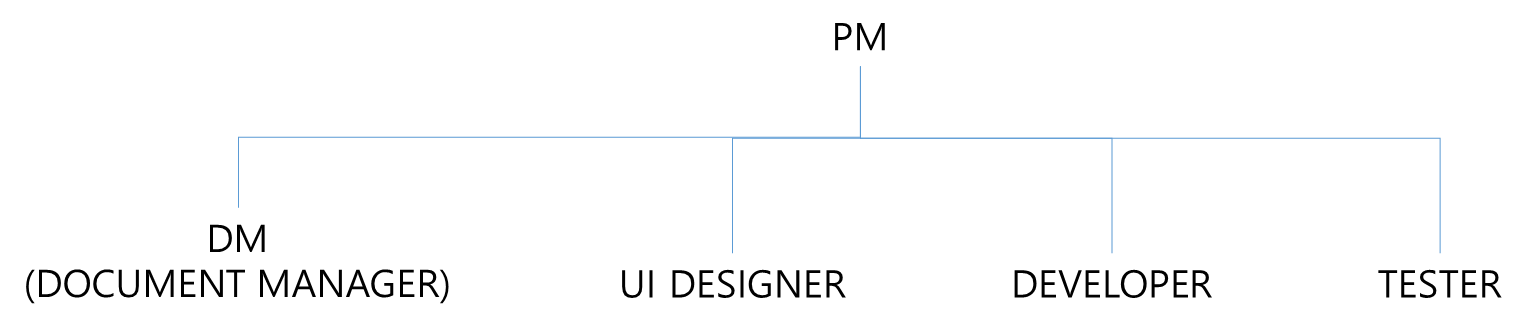
* 대학생들의 강의를 체계적으로 관리
* 간편한 조작을 통한 학습능률 향상

## 주요 일정

|  |  |  |
| --- | --- | --- |
| 단계 | 일정 | 산출물 |
| 프로젝트 계획서 초안 작성 | 3/28-3/30 (3) | 프로젝트 계획서 초안 |
| 요구사항 명세서 작성 | 3/31-4/8 (9) | 요구사항 명세서 |
| 프로젝트 계획서 작성 | 3/28-4/13 (17) | 프로젝트 계획서 |
| UI 디자인 | 4/30-5/27 (28) | UI 디자인 설계서 |
| 기능 구현 | 4/30-5/27 (28) | 소스 코드 |
| 테스트 | 5/28-6/1 (5) | 테스트 결과 보고서 |

## 조직

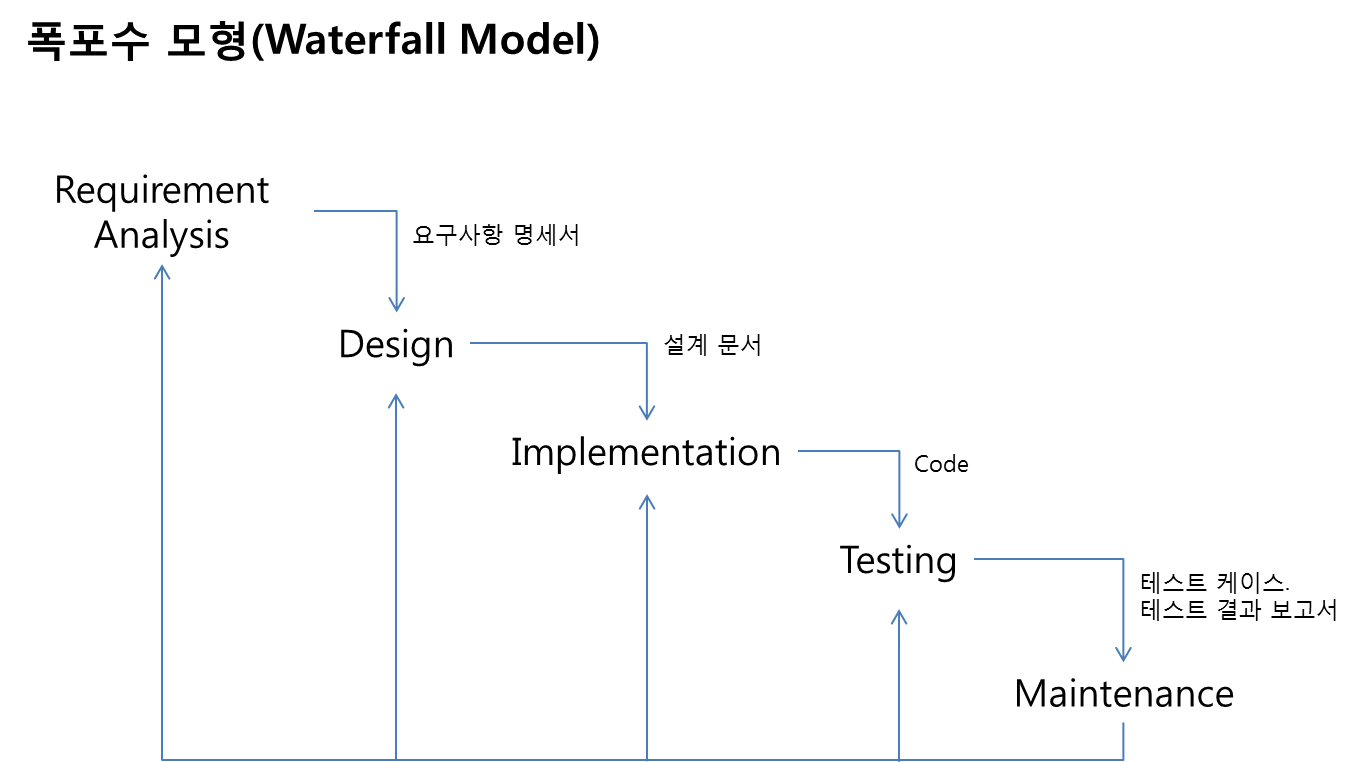
### 조직도



### 역할 및 책임

|  |  |  |
| --- | --- | --- |
| 팀원 | 역할 | 책임 |
| 김인환 | P.M (PROJECT MANAGER) | 전체적인 프로젝트 관리 |
| 김정현 | D.M (DOCUMENT MANAGER) | 프로젝트 문서화 |
| 남수완 | UI DESINGER | 사용자의 편의성에 맞춘 UI 디자인 및 CLASS 설계 |
| 우건희 | DEVELOPER | 발주자의 요구에 맞춘 기능 구현 |
| 김우빈 | TESTER | 결과물 테스트를 통한 오류제어 및 안정성 확보 |

## 생명주기 모델



개발 기술에는 크게 “Big Bang” approach 방식과 “Incremental” approach 방식이 있는데 Waterfall Model은 “Big Bang” approach 방식에 해당된다.

Waterfall Model은 Top-Down 접근 방식을 사용하는 개발 기술 중 하나로 계속해서 떨어지는 폭포수처럼 각 단계를 완전히 끝내고 다음 단계로 넘어가는 방식을 말한다.

그렇기 때문에 요구사항이 변동될 경우에 어려움이 생길 수 밖에 없다.

이를 방지하기 위해 요구사항을 빠트리지 않도록 WBS를 작성하고 Gantt 차트와 PERT 차트를 통해 일정계획을 세밀하게 작성할 계획이다. 또한 개발 단계에서 수정이 용이하도록 코드를 작성할 예정이다.

## 도구

**Eclipse**

-IDE (Integrated Development Environment)

-개발 환경을 구축한다.

**Maven3**

-Build System

동일한 빌드 구조 생성

Maven의 기본 생성 구조 준수

**Redmine**

-이슈 관리/추적 시스템 (Issue Management/Tracking System)

요구사항, 결함 등록 및 관리

**Git**

-분산 버전 관리 시스템 (Distributed Version Control System)

문서(요구사항 명세서, 프로젝트 계획서, 설계서, 테스트 케이스, 테스트 결과 보고서)와

소스코드의 버전 관리

파일의 최신 버전뿐만 아니라, 이전 버전까지의 전체 이력을 기록

**\*)GitHub**

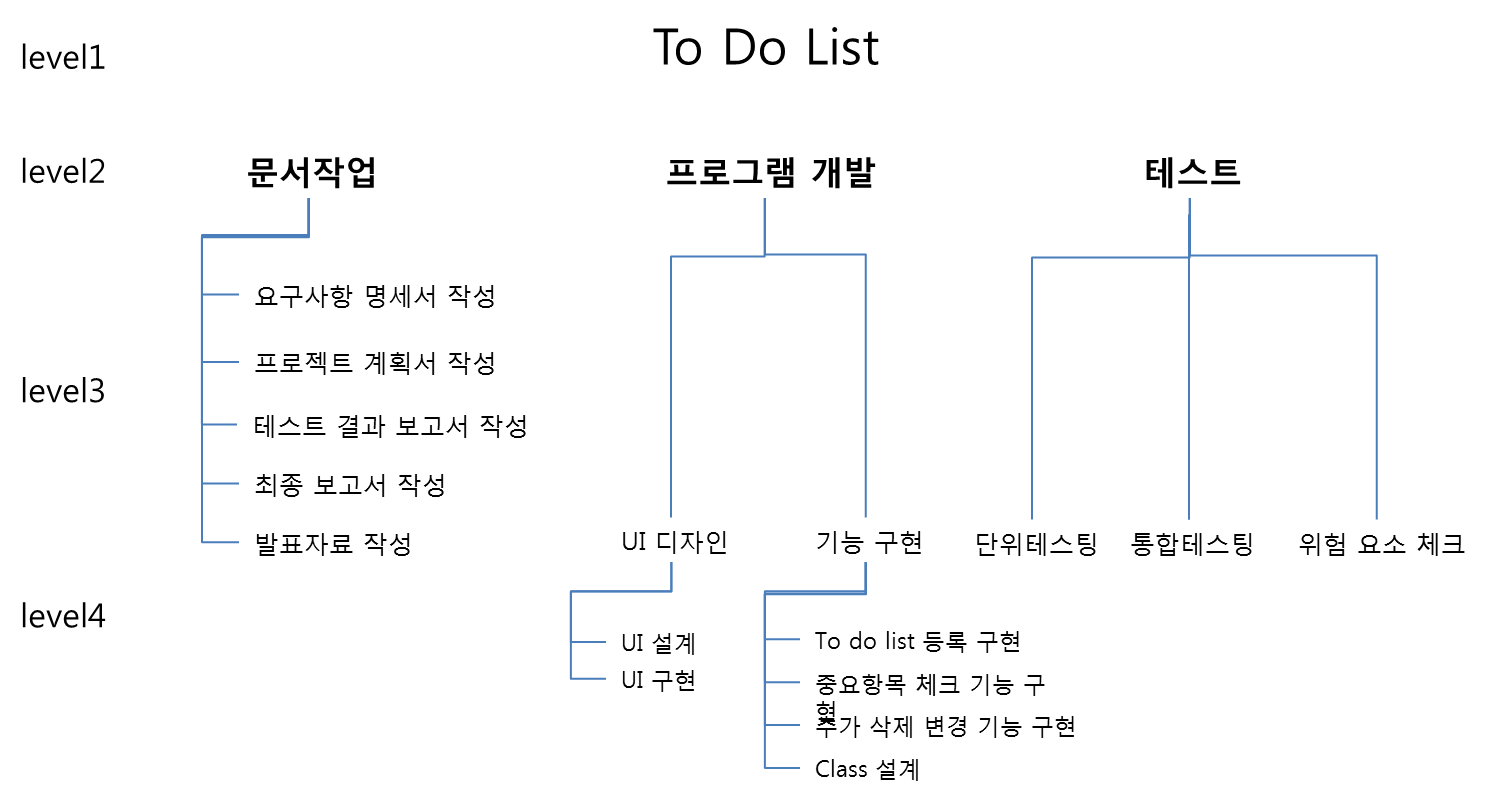
원격 저장소 (Remote Repository)

Repository를 공유할 수 있는 중앙 저장소

Git을 사용하는 프로젝트에게 저장소(repository)를 지원

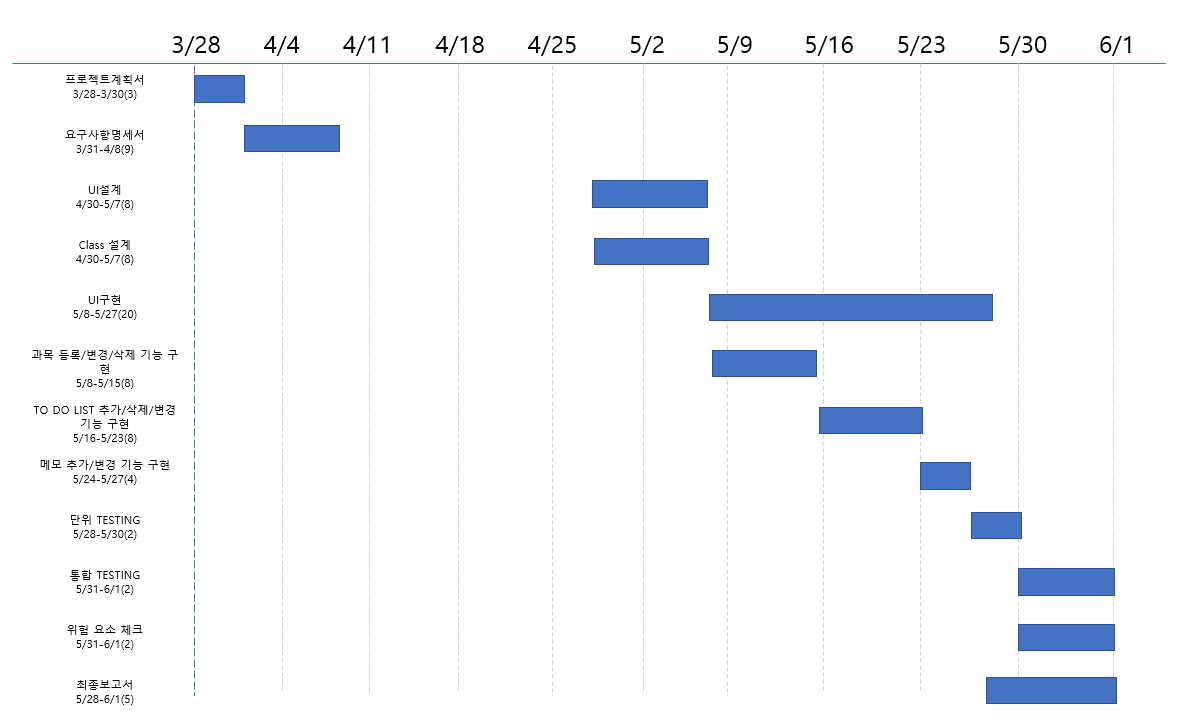
# 규모 산정

## WBS(Work Breakdown Structure)

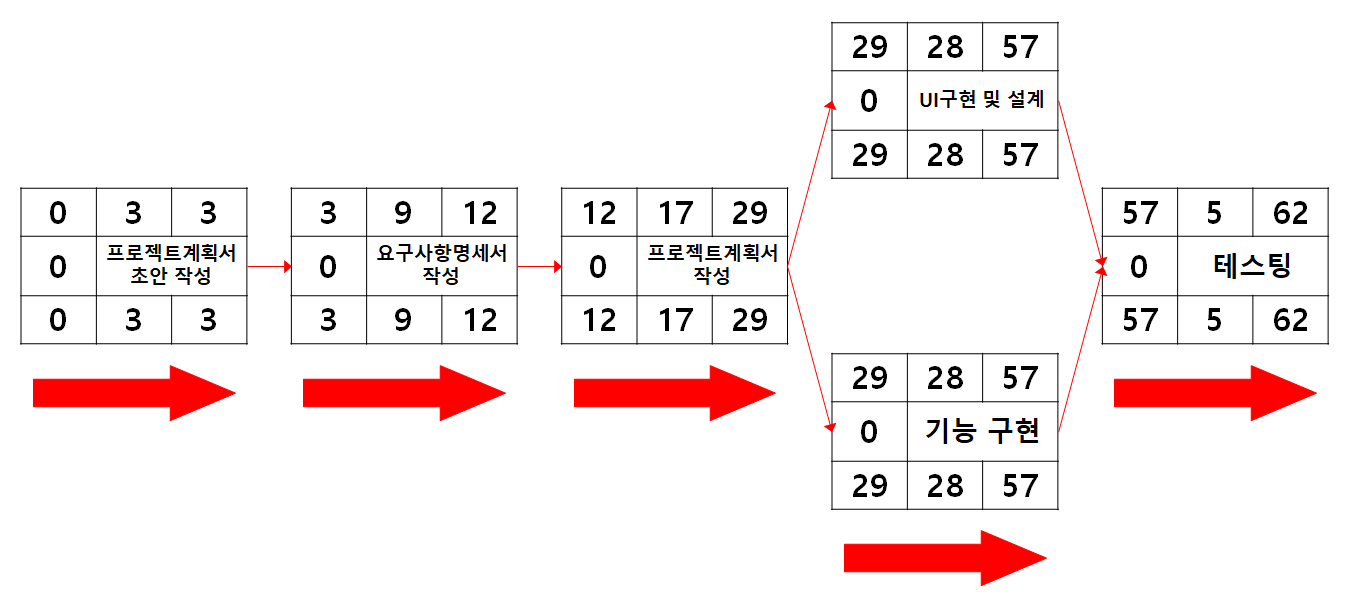


# 일정

**Gantt chart**



**PERT chart**



# 산출물 관리

**GitHub**

프로젝트의 모든 산출물은 GitHub repository에 저장한다.

<https://github.com/SMU-SE-Admin/18-1_3_THREEGO_Doc>

doc – 소스코드를 제외한, 나머지 작업 산출물들의 작업이 이루어지는 곳

요구사항 명세서, 프로젝트 계획서, Class 설계서/UI 설계서,

테스트케이스/테스트 결과 보고서,최종 보고서,

발표자료, 회의록 등 모든 문서를 Doc 파일로 저장한다.

<https://github.com/SMU-SE-Admin/18-1_3_THREEGO_Src>

src- 소스코드의 작업이 이루어지는 곳

프로그램 개발에 사용된 모든 소스코드를 저장한다.

**산출물 명**

문서 이름\_조 번호\_조 이름 .doc

월\_일\_조 이름\_회의록.doc

클래스 이름.java

**Commit Message**

작성 방법

-#Redmine일감번호 제목

본문  
작성 규칙

-제목과 본문을 한 줄 띄어 분리하여 작성

-제목은 20자 이내로 작성

-제목 끝에 . 금지

-제목은 명령조로 작성

-본문은 30자마다 줄 바꾸기

**버전 관리**

변동사항은 commit message작성 규칙에 따라 commit message를 작성하고, commit and push 하여 관리한다.

# 리스크 관리 계획

|  |  |  |  |
| --- | --- | --- | --- |
| 리스크 요소 | 가능성 | 영향도 | 대처 방안 |
| 요구사항 변동 | 상 | 상 | 개발과정에서 요구사항이 추가, 변동 되어도 기존 코드에 영향이 최소화되고 대처가 용이하도록 개발한다. |
| 팀원 이탈 | 하 | 중 | 불화로 인한 이탈은 발생하지 않도록 조원간의 의사소통을 많이 하고 부득이한 사정으로 이탈이 발생할 경우 PM은 다른 조원들에게 해당 업무를 적절하게 배분할 수 있도록 한다. |
| 개발 능력 부족 | 상 | 중 | 역할 분담을 가장 자신 있는 분야로 배분하고 담당 분야에 대한 부족한 지식을 다른 조원과 서로 보완해간다. |
| 개발 기간 부족 | 중 | 상 | 리스크 발생 우려가 많은 부분에 여유 기간을 주어 계획에 차질이 없도록 한다. |