프로젝트 계획서

<ONE>

|  |  |
| --- | --- |
| 컴퓨터과학과 201610996 | 박주원 |
| 컴퓨터과학과 201311088 | 권이안 |
| 컴퓨터과학과 201611026 | 이은선 |
| 컴퓨터과학과 201111166 | 이지홍 |
| 컴퓨터과학과 201611037 | 조금비 |

- 변 경 이 력 -

|  |  |  |  |
| --- | --- | --- | --- |
| 일자 | **버전** | **변경 내역** | **작 성 자** |
| 2018.3.20 | 1.0 | 프로젝트 산출물, 테스트계획, 요구사항 정리 작성 | 박주원 |
| 2018.3.22 | 1.1 | 프로젝트 조직도, 책임 및 역할, 프로젝트 환경, 리스트 관리 계획 작성 | 조금비 |
| 2018.3.23 | 1.2 | WBS , 일정 작성 | 이지홍 |
| 2018.3.25 | 1.3 | 간트, 퍼트차트 작성 | 권이안 |
| 2018.3.28 | 1.4 | 프로젝트 목적 작성, 전체적인 부분 다듬고 정리 | 이은선 |
| 2018.4.12 | 1.5 | 프로젝트 계획서 점검 및 보완 | 박주원 |
|  |  |  |  |

- 목 차 -

**1. 프로젝트 개요**

1.1 목적

1.2 주요 일정

1.3 조직

1.4 생명주기 모델

1.5 도구

**2. 규모 산정**

2.1 WBS

**3. 일정**

3.1 주요 일정

3.2 상세 일정

**4. 산출물 관리**

**5. 리스크 관리 계획**

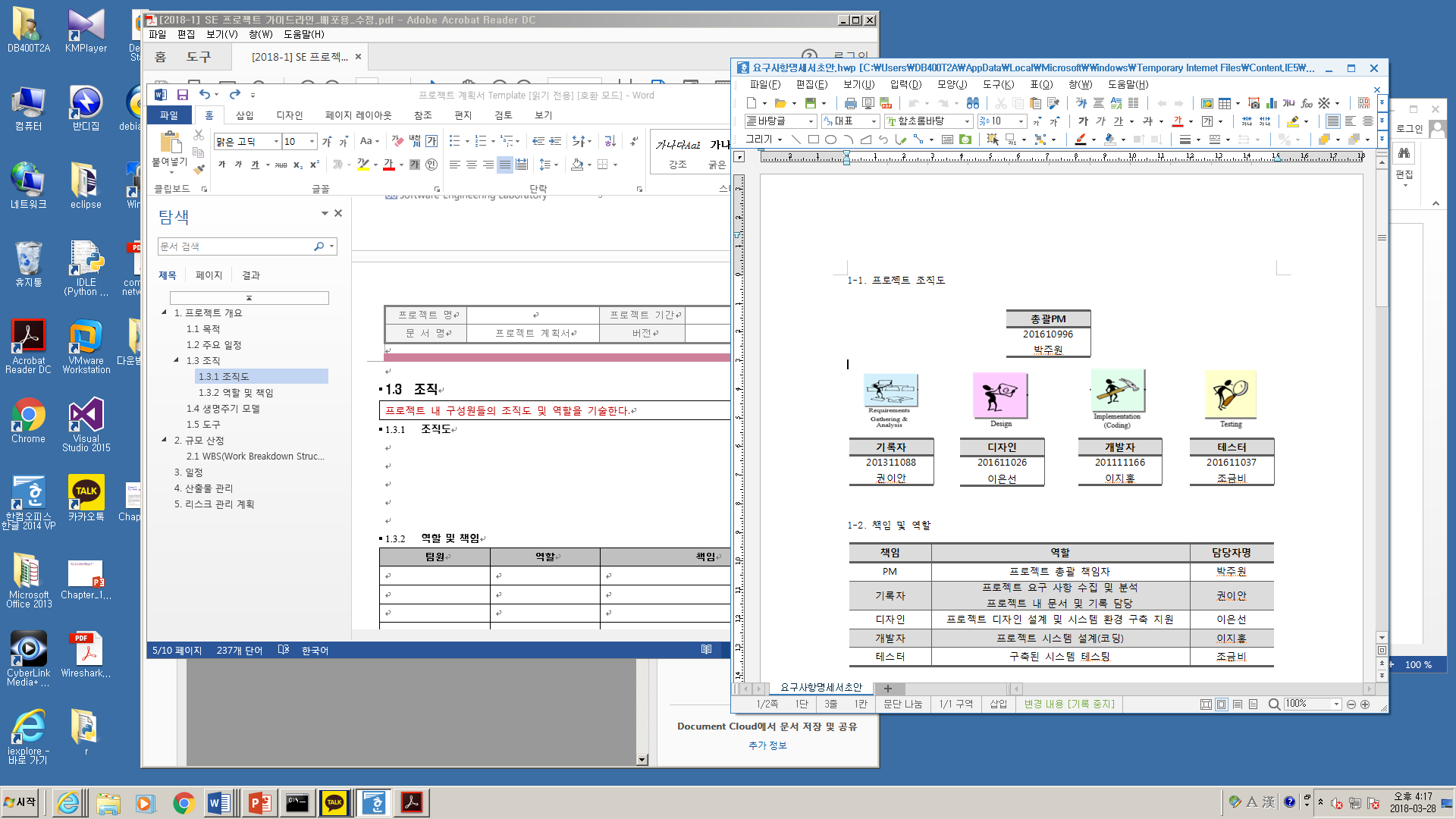
1. 프로젝트 개요
   1. 목적

|  |
| --- |
| 대학생이 수강하는 과목에 해당하는 To do(해야할일)를 등록하고 관리하는 프로그램을 개발함으로써 개발능력보다는프로젝트진행능력함양을 기른다. 실제 프로젝트를 진행하며 소프트웨어공학이론을 적용하면서 협업방법 및 요령을 터득하고 실무와 동일한 도구 사용 경험을 취득한다. 요구사항명세서(Software Requirements Specification,SRS), 프로젝트계획서(ProjectPlan), 설계서(Design Documents), Class 설계서(Class Design), UI설계서(UI Design), 테스트케이스 및 테스트결과보고서(TestCase &TestResultReport), 소스코드(SourceCode), 회의록을 개발 산출물로 제출한다. |

* 1. 주요 일정

|  |  |  |
| --- | --- | --- |
| **단계** | **작업** | **산출물** |
| 프로젝트 계획 | 프로젝트 범위 확정  프로젝트 일정 확정  프로젝트 진행 방향 확정 | 프로젝트 계획서  WBS |
| 시안 | 디자인 시안 | UI디자인, 시안 |
| 분석/설계 | 요구사항 분석 / 정의 / 관리  디자인가이드 제작  Class 설계 | 요구사항명세서  디자인가이드  Class 설계서 |
| 개발 | 기능 설계  프로그램 설계  화면 및 프로그램 설계  업무프로세스 설계  코드 설계  개발자 단위테스트 | 기능정의서  디렉토리구조 정의  프로그램사양서  업무흐름도  코드정의서  단위 테스트 결과 |
| 테스트 및 실행 | 통합테스트 및 미비점 보완 | 테스트 케이스(Test Case) & 테스트 결과 보고서(Test Result Report) |
| 완료 | 프로젝트 완료 | 프로젝트 완료보고서 |

* 1. 조직
     1. 조직도



* + 1. 역할 및 책임

|  |  |  |
| --- | --- | --- |
| **책임** | **역할** | **담당자명** |
| PM | 프로젝트 총괄 책임자 | 박주원 |
| 기록자 | 프로젝트 요구 사항 수집 및 분석  프로젝트 내 문서 및 기록 담당 | 권이안 |
| 디자인 | 프로젝트 디자인 설계 및 시스템 환경 구축 지원 | 이은선 |
| 개발자 | 프로젝트 시스템 설계(코딩) | 이지홍 |
| 테스터 | 구축된 시스템 테스팅 | 조금비 |

* 1. 생명주기 모델

|  |
| --- |
|  |

* 1. 도구

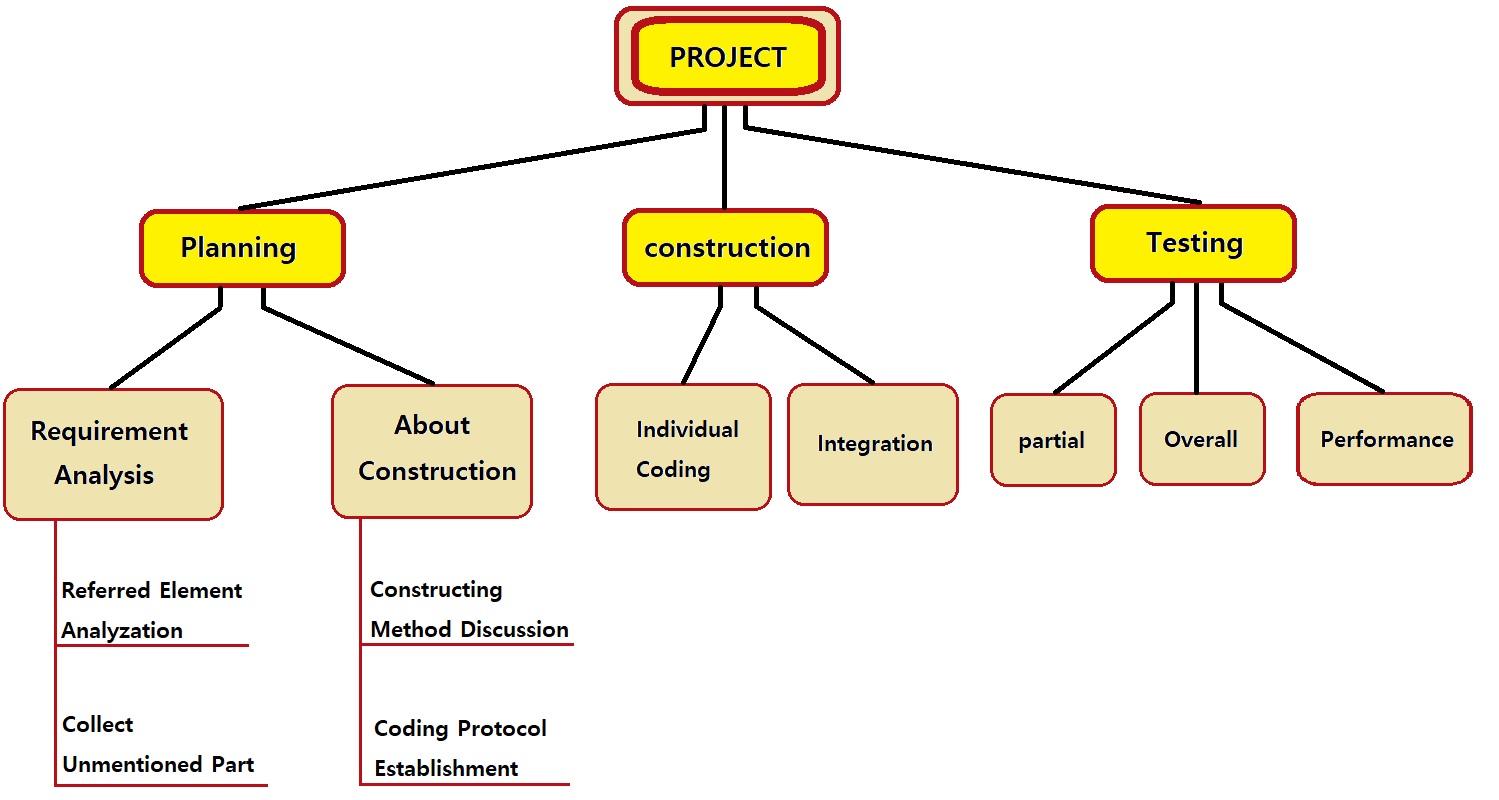
(1) 하드웨어 자원

|  |  |  |
| --- | --- | --- |
| **구분** | **TYPE** | **SPEC** |
| Server | CPU | intel CORE i5 |
| Memory | 8GB |
| HDD | 256GB |

(2) 소프트웨어 자원

|  |  |  |
| --- | --- | --- |
| **구분** | **TYPE** | **SPEC** |
| Server | OS | Windows 10 Home |
| JDK | 1.8 |
| Tool | Life Cycle : Waterfall Model |
| IDE : Eclipse |
| Build System : Maven3 |
| IM/TS : Redmine |
| DVCS : Git |

1. 규모 산정
   1. WBS(Work Breakdown Structure)



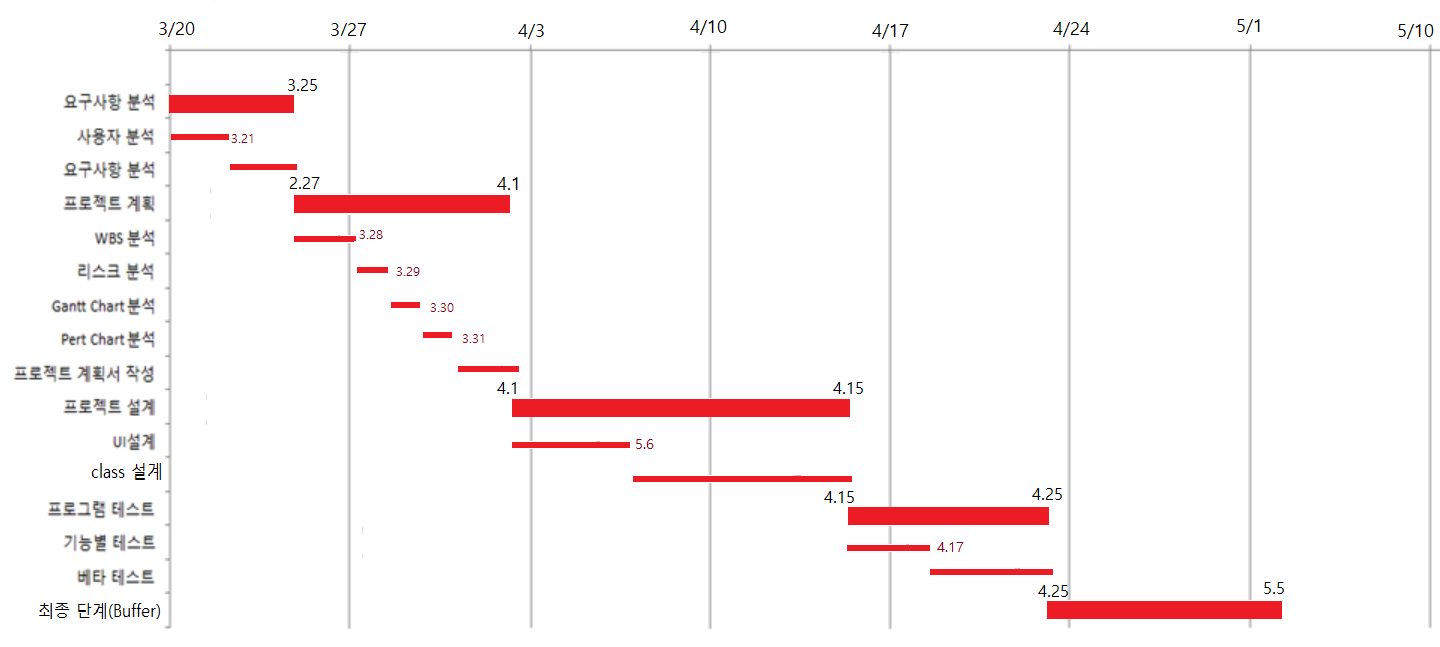
1. 일정
   1. 주요일정

|  |  |  |
| --- | --- | --- |
| 단계 | **일정** | **산출물** |
| 요구사항 분석 | 2018.03.20~03.27 | 요구사항 명세서 |
| 프로젝트 계획 | 2018.03.27~03.30 | 프로젝트 계획서 |
| 프로그램 설계 | 2018.04.01.~04.15 | 설계서 |
| 프로그램 테스트 | 2018.04.15.~04.22 | 테스트 결과 보고서 |
| 최종 단계(Buffer) | 2018.04.29~05.10 | 소스코드 및 최종 프로그램 |

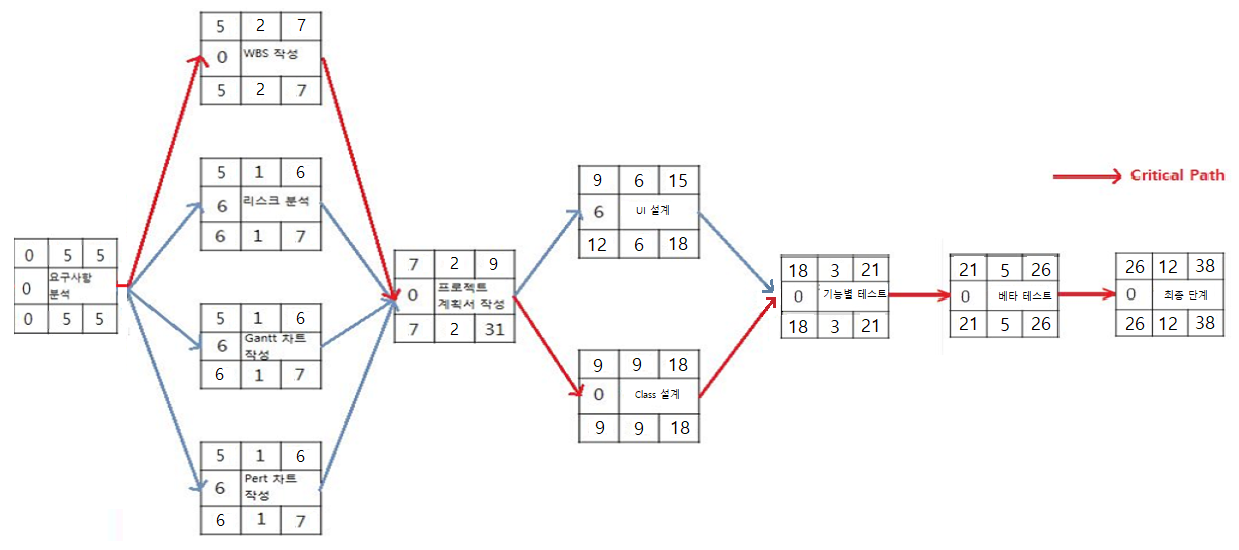
* 1. 상세 일정

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | 작업명 | 시작일 | 종료일 | 기간 |
| 1 | 요구사항 분석 | 2018-03-20 | 2018-03-25 | 5 |
| 1.1 | 사용자 분석 | 2018-03-20 | 2018-03-21 | 2 |
| 1.2 | 요구사항 분석 | 2018-03-21 | 2018-03-23 | 2 |
| 2 | 프로젝트 계획 | 2018-03-27 | 2018-04-01 | 7 |
| 2.1 | WBS 분석 | 2018-03-27 | 2018-03-28 | 2 |
| 2.2 | 리스크 분석 | 2018-03-28 | 2018-03-29 | 1 |
| 2.3 | Gantt 차트 분석 | 2018-03-29 | 2018-03-30 | 1 |
| 2.4 | Pert 차트 분석 | 2018-03-30 | 2018-03-31 | 1 |
| 2.5 | 프로젝트 계획서 작성 | 2018-03-31 | 2018-04-01 | 2 |
| 3 | 프로그램 설계 | 2018-04-01 | 2018-04-15 | 15 |
| 3.1 | UI 설계 | 2018-04-01 | 2018-05-06 | 6 |
| 3.2 | Class 설계 | 2018-04-06 | 2018-04-15 | 9 |
| 4 | 프로그램 테스트 | 2018-04-15 | 2018-04-25 | 8 |
| 4.1 | 기능별 테스트 | 2018-04-15 | 2018-04-17 | 3 |
| 4.2 | 베타 테스트 | 2018-04-17 | 2018-04-22 | 5 |
| 5 | 최종 단계(Buffer) | 2018-04-25 | 2018-05-05 | 12 |
|  | 합계 | 2018-03-20 | 2018-05-05 | 81 |

**Gantt Chart**



**Pert Chart**



1. 산출물 관리
2. **1항. 덮어쓰기를 금한다.**
3. **2항. java확장자 파일을 통해 버전관리를 한다.**
4. **3항. 코드 마지막에 버전별 변경이력을 주석처리를 기록하여 누적시킨다.**
5. **4항. java확장자 파일명은 “버전명\_최종수정자명”으로 한다.**
6. **5항. 수정사항 발생(요구사항 변경)시 회의를 통하는 것을 원칙으로 한다.**
7. **6항. 모든 산출물은 공유 가능한 클라우드 1개 계정에 모은다.**
8. 리스크 관리 계획

**가능성**

리스크가 발생할 수 있는 확률(0%~100%)

**영향도**

**상** : 리스크가 발생했을 때, 해당 단계에서 해결이 불가하고 전 단계로 이동해야 하는 상황

**중** : 리스크가 발생했을 때, 해당 단계에서 해결이 가능한 상황

**하** : 리스크가 발생했을 때, 해당 단계에서 짧은 시일 내에 해결이 가능한 상황

