(산출물양식) 단위테스트 계획서

|  |
| --- |
| 단위테스트 계획서 |

프로젝트 명

2017.08.31

에이블맥스 ㈜

목차

[I. 단위테스트 개요 2](#_Toc492388010)

[II. 테스트 수행 2](#_Toc492388011)

[1. 테스트 절차 2](#_Toc492388012)

[2. 업무별 담당자 3](#_Toc492388013)

[III. 테스트 케이스와 주요 점검사항 3](#_Toc492388014)

[1. 테스트 케이스 도출 3](#_Toc492388015)

[2. 테스트 주요점검 사항 4](#_Toc492388016)

[IV. 테스트 조직과 수행일정 6](#_Toc492388017)

[1. 테스트 조직과 역할 6](#_Toc492388018)

[2. 테스트 수행일정 6](#_Toc492388019)

[3. 테스트 결함관리 7](#_Toc492388020)

# 단위테스트 개요

단위테스트는 단위 프로그램이나 모듈이 프로그램 명세서에 정의된 바와 같이 기능의 오류 없이 수행되는지를 검증하는 테스트로 개발자 중심의 테스트이다. 단위테스트는 사용자나 운영자가 인식할 수 있는 가장 작은 단위의 기능이 시스템명세서 (화면 프로그램 설계서, 프로그램설계서 등)에 따라 구현되었는지를 확인하는데 그 목적이 있다.

단위테스트를 수행함으로써 아래와 같은 목표를 달성할 수 있다.

* 각 업무별 단위프로그램의 정확성 확인
* 단위 기능구현의 완전성 확인
* 통합테스트의 사전 기반 조성

# 테스트 수행

## 테스트 절차

일반적인 단위 테스트 절차는 다음과 같다. 단위테스트 절차는 해당 프로젝트의 규모와 성격에 따라 다를 수 있으며, 프로젝트 수행의 효율성과 효과성을 고려하여 적절한 수행절차를 변경할 수 있다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **구분** | **QA(PM)** | **PL** | **개발자** | **기반/전환** | **산출물** |
| 단위테스트 계획 | *단위테스트 계획 검토*  *단위테스트 계획*  *단위테스트 수행 방법 숙지* |  |  |  | .단위테스트 계획서 |
| 단위테스트 설계 |  | *단위테스트 설계* |  |  | . 테스트 시나리오(Optional)  . 테스트 케이스 |
| 단위테스트 실행 |  | *단위테스트 실행* | **Y**  결함 수정  결함 |  | .테스트 케이스/로그 |
| 단위테스트 평가 | *결과 보고서 작성* |  |  |  | .단위테스트 결과서 |

## 업무별 담당자

테스트를 수행하기 위해 책임과 역할을 나누게 되며, 해당 업무별 담당자를 설정한 예가 아래와 같이 표시되어 있다. 이 또한 위의 절차를 어떻게 수정하여 활용하는가에 따라 변경할 수 있다.

|  |  |  |  |
| --- | --- | --- | --- |
| **역할** | **담당** | **활동** | **비고** |
| 시험총괄(PM) |  | 단위테스트 관리 및 조정 |  |
| QA |  | 표준부합 여부 확인 |  |
| PL/개발자 |  | 테스트 케이스, 테스트 시나리오 작성 |  |
| PL/개발자 |  | 테스트 케이스/로그 작성 |  |
| PM |  | 결과 보고서 |  |

# 테스트 케이스와 주요 점검사항

## 테스트 케이스 도출

단위 테스트 설계 시 단위테스트 케이스를 도출하기 위한 전략은 다음과 같다.

* 제공된 테스트 케이스 샘플을 기반으로 테스트 케이스 설계가 수행되도록 지원한다.
* 테스트 케이스 도출 시 사용자가 시스템을 올바르게 사용하는 경우(positive test case) 뿐만 아니라 예상되지 않는 행위를 하는 경우(negative test case)도 고려하여 테스트 케이스를 작성한다.
* 개발 산출물에 대한 테스트 커버리지를 적용하여 시스템의 어느 정도가 테스트 되었는지를 측정 가능하게 한다.

## 테스트 주요점검 사항

단위 테스트 수행 시 점검해야 할 주요 사항은 다음과 같다.

* **사용자 인터페이스**

|  |  |
| --- | --- |
| **No** | **점검 항목** |
| 1 | 프로젝트 표준에 따라 GUI가 설계되어 있는지 확인하는가? |
| 2 | 탭 시퀀스 기능이 제대로 동작하는지 확인하는가? |
| 3 | 입력 필드에 커서가 제대로 나타나는지 확인하는가? |

* **필드 유효성 – 문자/ 숫자 /날짜/시간 필드**

|  |  |
| --- | --- |
| **No** | **점검 항목** |
| 1 | 입력 필드에 대한 필드 길이를 확인하는가? |
| 2 | 공백, NULL값, 특수문자와 같은 이 입력값에 대한 확인을 하는가? |
| 3 | 경계값(boundary condition) 확인을 하는가? |
| 4 | 날짜 필드에서 형식에서 벗어난 입력 값을 확인하는가? |
| 5 | 숫자 필드의 절사 여부를 확인하는가? |

* **에러 메시지**

|  |  |
| --- | --- |
| **No** | **점검 항목** |
| 1 | 작성된 모든 에러 메시지를 확인하는가? |
| 2 | 에러 메시지의 의미와 적절성을 확인하는가? |
| 3 | 복수개의 에러를 시뮬레이션할 수 있는가? |
| 4 | 에러 메시지가 출력된 후 적절할 필드에 커서가 위치하는지 확인하는가? |
| 5 | 필수 입력 필드 누락에 대한 에러 메시지를 확인하는가? |

* **기능성 – 검색/프로세스 로직/데이터베이스**

|  |  |
| --- | --- |
| **No** | **점검 항목** |
| 1 | 조건 커버리지 혹은 경로 커버리지를 확인하는가? |
| 2 | 데이터베이스의 첫번째와 마지막 레코드 검색에 대한 기능을 확인하는가? |
| 3 | 데이터베이스에 존재하지 않는 레코드에 대한 검색을 확인하는가? |
| 4 | 검색어가 공백, 부분 입력 데이터, 전체 입력 데이터일 경우 검색 기능을 확인하는가? |
| 5 | 0을 제수로 한 나눗셈 연산을 확인하는가? |
| 6 | 최대값이나 최소값에 대한 연산 처리를 확인하는가? |
| 7 | 입력, 수정, 삭제와 같은 데이터베이스 트랜잭션을 확인하는가? |
| 8 | 주키, 외래키의 제약 조건을 이용해 데이터베이스 운영을 확인하는가? |
| 9 | 에러 핸들링을 확인하는가? |

* **보고서 출력**

|  |  |
| --- | --- |
| **No** | **점검 항목** |
| 1 | 컨텐츠의 머리글과 바닥글을 확인하는가? |
| 2 | 보고서에서 페이지 합계 및 전체 합계의 정확성을 확인하는가? |
| 3 | 프린터 구성요소와 보고서 레이아웃을 확인하는가? |
| 4 | 값이 없거나 최대 입력값을 가지고 있는가를 확인하는가? |

# 테스트 조직과 수행일정

## 테스트 조직과 역할

테스트 수행을 위해 테스트 수행조직을 구성하고 각자의 역할을 정의한다.

|  |  |  |
| --- | --- | --- |
| **역할** | **담당자** | **책임** |
| 품질 관리자 |  | 고객 협력 유도, 테스트 관련 인적/물적자원 조달이 가능하도록 지원, 테스트 결과 평가, 테스트 결과 집계 및 통계 작성 |
| 테스트 담당자 |  | 테스트 계획서, 테스트 케이스 작성 가이드, 테스트 수행 총괄, 테스트 관련 산출물 확인, 테스트 결과서 작성 |
| 테스터 |  | 테스트 케이스 작성, 테스트 데이터 준비  단위테스트 수행  테스트 과정 중 파악된 결함 수정 |

## 테스트 수행일정

단위 테스트 단계의 주요 수행 일정은 다음과 같이 수립할 수 있으며, 아래의 예를 참조하여 테스트 일정을 수립한다. 테스트 일정에 대한 공유는 워크샵 혹은 별도의 교육일정을 수립하여 전파하고 필요한 경우 주요 테스트 도구에 대한 교육도 동시에 진행하는 것이 좋다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **구분** | **주요 활동** | **일정** | | **비고** |
| 단위  테스트 | 단위테스트 계획서 보완 | 05.21 ~ 5.30 | | . 단위 테스트 수행을 위한 단위테스트 계획서 상세화 |
| 단위테스트케이스 BP 도출 | 06.01 ~ 6.14 | | . 단위 테스트 케이스 및 데이터 샘플 작성 |
| 단위 테스트 케이스 작성 | 업무 | 06.11-08.27 | 단위 테스트 케이스 및 데이터의 적절성 검토  . 테스트 대상 서버(개발 or 운영)에 소스 및 DB 이관  . 단위 테스트 수행 결과 작성 |
| 테스트데이터 및 드라이버/스텁 준비 | 업무 | 06.25-10.25 |
| 단위 프로그램 별 테스트 수행 | 업무 | 06-25-11.08 |
| 식별된 결함 제거 및 프로그램 수정/보완 | 08.05~11.08 | |
| 단위 테스트 결과서 작성 및 보고 | 08.05~11.08 | |
| PL 현업 TFT 검증 | 단위 테스트 수행 결과 검증 | 08.05~11.08 | | . 단위 테스트 수행 결과 검증 |
| 단위 테스트결과서 검토 | 11.15 | | . 단위 테스트 종료 여부에 대한 의사 결정  . 합의한 기준에 따라서 통합 테스트 진입여부 결정 |

## 테스트 결함관리

단위 테스트 단계에서 발견된 결함을 결함 관리를 위해 '결함관리대장' 양식에 기록하고 관리하거나 별도의 테스트관리도구를 활용하여 자동으로 각종 결함이 등록될 수 있도록 하는 것이 필요하다. 결함관리를 위한 대표적인 도구로 JAMA, Confluence 등이 있다. 만약 별도의 도구를 도입할 수 없는 상황이라면 결함관리대장을 작성하여 엑셀로 관리한다.

결함관리대장을 프로젝트 리더는 결함 결과와 이슈를 프로젝트 관리자에게 보고한다. 결함에 대한 관리는 계량적으로 관리할 필요가 있으며, 계량적 관리는 지표를 설정하고 주기적으로 지표를 계산하여 추이를 분석하는 것이 필요하다. 대표적인 지표는 테스트 수행진척율, 결함율, 결함해소율이다.

|  |  |  |  |
| --- | --- | --- | --- |
| **지표** | **A** | **B** | **계산** |
| 테스트 진척율 | 테스트 완료된 케이스 수 | 전체 테스트 케이스 수 | = A / B X 100% |
| 결함율 | 결함 테스트 케이스 수 | 테스트 완료된 케이스 수 | = A / B X 100% |
| 결함해소율 | 해결된 결함수 | 결함 테스트 케이스 수 | = A / B X 100% |