CLOUD LAB MASTER TEST PLAN

Version Information

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Remarks** | **Author** |
| 0.1 | 09/04/2021 | Initial Draft | ITC303 Team 11 |
| 1.0 | 11/04/2021 | Initial Version | ITC303 Team 11 |

**Project objective**

The objective of the Cloud Lab System is to provide a drag-and-drop interface where you create and configure a lab, with the ability to make multiple copies of the lab to accommodate the number of students who will be using it. Using AWS cloud services and web access via a desktop or laptop.

**Test objectives**

* Availability – To verify that the system is capable of maintaining satisfactory availability.
* Persistence - To verify the system can store and retrieve data effectively.
* Reliability – To verify that the system will remain operational under adverse conditions.
* Scalability – To verify that the system will adequately scale as the user base grows.
* Responsiveness – To verify that the systems maximum response time is within specification.
* Security – To verify that the user's personal information is adequately encrypted.
* Authentication – To verify that only authorized users can access the system.
* Usability – To verify that the system and its interfaces are easy to use and do not hinder operation of the system.

**Test approach**

* Unit testing: Security and Authentication
* Integration testing: Persistence, Security and Authentication
* System testing: Availability, Persistence, Reliability, Scalability and Responsiveness
* Acceptance Testing: Availability, Reliability, Scalability, Responsiveness, Security, Authentication and Usability

Table of Contents

[1](#_heading=h.gjdgxs) Introduction 4

[1.1](#_heading=h.30j0zll) Project and project objective 4

[1.2](#_heading=h.1fob9te) Objective of the master test plan 4

[2](#_heading=h.3znysh7) Documentation 5

[2.1](#_heading=h.2et92p0) Basis for the master test plan 5

[2.2](#_heading=h.tyjcwt) Test basis 5

[3](#_heading=h.3dy6vkm) Test strategy 6

[3.1](#_heading=h.1t3h5sf) Risk analyses 6

[3.1.1](#_heading=h.4d34og8) Product Risk Analysis 6

[3.1.2](#_heading=h.2s8eyo1) Technical Risk Analysis 7

[3.2](#_heading=h.3rdcrjn) Test strategy 8

[4](#_heading=h.lnxbz9) Test Levels 9

[4.1](#_heading=h.44sinio) The <name test level> 9

[4.1.1](#_heading=h.z337ya) Entrance and Exit Criteria 9

[4.1.2](#_heading=h.3j2qqm3) Test Environment 9

[4.1.3](#_heading=h.1y810tw) Test Objectives 9

# Introduction

## Project and project objective

The objective of the Cloud Lab System is to provide a drag-and-drop interface where you create and configure a lab, with the ability to make multiple copies of the lab to accommodate the number of students who will be using it. Using AWS cloud services and web access via a desktop or laptop.

## Objective of the master test plan

The objective of the Master Test Plan (MTP) is to inform all who are involved in the test process about the approach, the activities, including the mutual relations and dependencies, and the (end) products to be delivered.

The master test plan describes the test approach, the activities and (end) products.

# Documentation

This chapter describes the documentation used in relation with the master test plan. The described documentation concerns a first inventory and will be elaborated, actualized and detailed at a later stage, during the separate test levels.

## Basis for the master test plan

The following documents are used as the basis for this master test plan.

|  |  |  |  |
| --- | --- | --- | --- |
| **Document name** | **Version** | **Date** | **Author** |
| **Cloud Lab System Vision.docx** | 1.0 | 11.04.21 | ITC303 Team 11 |
| **Non-Functional Requirements Specification.docx** | 1.0 | 11.04.21 | ITC303 Team 11 |
| **Cloud Lab Risk List.docx** | 1.0 | 11.04.21 | ITC303 Team 11 |

## Test basis

The test basis contains the documentation that serves as the basis for the tests that have to be executed. The overview below describes the documentation that is the starting point for testing.

|  |  |  |  |
| --- | --- | --- | --- |
| **Document name** | **Version** | **Date** | **Author** |
| **Critical Core Use Cases.docx** | 1.0 | 23.11.19 | ITC303 Team 11 |
| **Domain Model** | 1.0 | 11.04.21 | ITC303 Team 11 |
| **Use Case Diagrams** | 1.0 | 11.04.21 | ITC303 Team 11 |
| **Architecture Notebook.docx** | 1.0 | 11.04.21 | ITC303 Team 11 |

# Test strategy

The time available for testing is limited; not everything can be tested with equal thoroughness. This means that choices have to be made regarding the depth of testing. Also, it is strived to divide test capacity as effectively and efficiently as possible over the total test project. This principle is the basis of the test strategy.

The test strategy is based on risks: a system has to function in practice to an extent that no unacceptable risks for the organization arise from it. If the delivery of a system brings along many risks, thorough testing needs to be put in place; the opposite of the spectrum is also true: 'no risk, no test'.

The first step in determining the test strategy is the execution of a product risk analysis. This is elaborated in §3.1.

The test strategy is subsequently based on the results of the risk analyses. The test strategy lays down what, how and when (in which test level) is being tested and is focused in finding the most important defects as early as possible for the lowest costs. This can be summarized as testing with an optimal use of the available capacity and time. The test strategy is described in §3.3.

## Risk analyses

### Product Risk Analysis

The product risks are determined in cooperation with the client and the other parties involved. Product risks are those risks associated with the final product failing to meet functional requirements and required system quality characteristics (NFRs) This product risk analyses (PRA) is comprised of two steps:

|  |  |  |  |
| --- | --- | --- | --- |
| **Product Risk** | **Characteristic** | **Description** | **Risk Class** |
| 1 | Usability | Making sure system does not become difficult to use | A |
| 2 | Persistence | Losing track of user information and session data | B |
| 3 | Availability | Losing functionality if the system down for maintenance or loses connection | C |
| 4 | Reliability | The system crashing of being shut down for repairs | D |
| 5 | Security | Unauthorized access to personal information or session data | E |
| 6 | Scalability | The user base growing too large and overloading the system | F |

The extent of the risk (the risk class) is dependent on the chance of failure (how big the chance is that it goes wrong?) and it depends on the damage for the organization if it actually occurs.

### Technical Risk Analysis

Technical risks are determined in cooperation with the analyst/designers and programmers involved. Technical risks are development risks associated with failing to create a system that behaves according to specifications derived from requirements. (I.E. those aspects of development that pose particular challenges.) This technical risk analyses (TRA) is comprised of two steps:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Technical risk** | | **Risk Area** | **Description** | **Risk Class** |
| 1 | Availability | | High availability requirements of the front and backend both online and offline | A |
| 2 | Persistence | | Protection against data loss | B |
| 3 | Reliability | | Reliability of the backend | A |
| 4 | Scalability | | Scalability of the backend | B |
| 5 | Responsiveness | | The system response time | B |
| 6 | Security | | Data privacy/encryption | B |
| 7 | Authentication | | User identification system | C |
| 8 | Usability | | Maintain ease of use | C |

## Test strategy

For each risk from the product and technical risk analysis the risk class determines the thoroughness of the test. Risk class A is the highest risk class and C the lowest. The test strategy is subsequently focused on covering the risks with the highest risk class as early as possible in the test project.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Risk | Description | Risk Cat | Test Level | | | | | |
| SR | Unit | Int | FAT | UAT | ST |
| Availability | High availability requirements of the front and backend both online and offline | A |  |  |  | \*\*\* | \*\* | \* |
| Persistence | Protection against data loss | B | \*\* |  | \*\*\* | \*\* |  | \*\* |
| Reliability | Reliability of the backend | A |  |  |  | \*\*\* | \*\* | \* |
| Scalability | Scalability of the backend | B |  |  |  | \*\* | \*\* | \*\* |
| Responsiveness | The system response time | B |  |  |  | \*\* | \*\* | \*\* |
| Security | Data privacy/encryption | B | \* | \* | \*\* | \*\* |  |  |
| Authentication | User identification system | C |  |  | \* | \*\* | \*\* | \* |
| Usability | Maintain ease of use | C | \*\* |  |  | \*\* | \*\*\* |  |

Legend for the table above:

|  |  |
| --- | --- |
| RC | Risk class (from product and technical risk analysis, where A=high risk, B=average risk, C=low risk) |
| SR | Static Review of the various intermediary products (requirements, functional design, technical design). Checking and examining artefacts without executing the software |
| Unit | Unit test and Unit integration test |
| Integration | Integration tests (low level (L), high level(H)) |
| FAT | Functional acceptance test (alpha stage UAT) |
| UAT | User acceptance test (Beta stage UAT) |
| ST | System test (functional scenario testing (F), system quality scenario testing (S)) |
| ⬤ | Limited thoroughness of the test |
| ⬤⬤ | Medium thoroughness of the test |
| ⬤⬤⬤ | High thoroughness of the test |
| <blank> | If a cell is blank, it means that the relevant test or evaluation level does not have to be concerned with the characteristic |

# Test Levels

For this MTP the following test levels are acknowledged:

|  |  |
| --- | --- |
| **Test level** | **Goal** |
| Unit testing: | The aim is to test each part of the software by separating it. It checks that component are fulfilling functionalities or not |
| Integration testing: | In this testing phase, different software modules are combined and tested as a group to make sure that the integrated system is ready for system testing. Integrating testing checks the data flow from one module to other modules. |
| System testing: | System testing is performed on a complete, integrated system. It allows checking the system's compliance as per the requirements. It tests the overall interaction of components. It involves load, performance, reliability and security testing. |
| Acceptance testing: | Acceptance testing is a test conducted to find if the requirements of a specification or contract are met as per its delivery. |

## The Unit Testing

The primary goal of Unit testing is to isolate each part of the program and demonstrate that the individual parts as they are supposed to.

### Entrance and Exit Criteria

### Test Environment

### Test Objectives

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Test Goals** | **Risk Verification** | **Schedule** |
| **Security** | To verify that user’s personal information is adequately encrypted |  | During implementation of |
| Authentication | To verify that only authorized users can access the system |  | During implementation of |

## 

## The Integration Testing

The primary goal of Integration testing is to combine individual components and test them as a group.

### Entrance and Exit Criteria

### Test Environment

### Test Objectives

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Test Goals** | **Risk Verification** | **Schedule** |
| **Persistence** | To verify the system can storage and retrieve data effectively |  | During implementation of |
| Security | To verify that user’s personal information is adequately encrypted |  | During implementation of |
| Authentication | To verify that only authorized users can access the system |  | During implementation of |

## The System Testing

The primary goal of System testing is to test and evaluate the complete system's compliance with the specified requirements.

### Entrance and Exit Criteria

### Test Environment

### Test Objectives

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Test Goals** | **Risk Verification** | **Schedule** |
| **Availability** | To verify that the system is capable of maintaining satisfactory availability |  | During implementation of |
| Persistence | To verify the system can store and retrieve data effectively |  | During implementation of |
| Reliability | To verify that the system will remain operational under adverse conditions such as crashes and hardware failure. |  | During implementation of |
| Scalability | To verify that the system will adequately scale as the user base grows |  | During implementation of |
| Responsiveness | To verify that the systems maximum response time is within specification |  | During implementation of |

## The Acceptance Testing

The primary goal of Acceptance testing is to verify whether or not the system has met the business requirements.

### Entrance and Exit Criteria

### Test Environment

### Test Objectives

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Test Goals** | **Risk Verification** | **Schedule** |
| **Availability** | To verify that the system is capable of maintaining satisfactory availability |  | During implementation of |
| Reliability | To verify that the system will remain operational under adverse conditions |  | During implementation of |
| Scalability | To verify that the system will adequately scale as the user base grows |  | During implementation of |
| Responsiveness | To verify that the systems maximum response time is within specification |  | During implementation of |
| Security | To verify that user’s personal information is adequately encrypted |  | During implementation of |
| Authentication | To verify that only authorized users can access the system |  | During implementation of |
| Usability | To verify that the system and its interfaces are easy to use and do not hinder operation of the system |  | During implementation of |