# 5. Selected Test Design Techniques

1. Test Strategy: model-based (authentication model)

2. Test strategy: analytical - based on risk - tests designed and prioritized on the level of risk.

3. Test strategy: process compliant  
Test dimensions: desired result

4. Test strategy: reactive  
Test dimensions: Scope (what gets tested), Testers (who does the testing - people who normally use the product)

Part I - Authentication - Black Box Testing

The **test strategy** is based on the **authentication model** of the application, the users should be able to register with a username and password and log-in using only the username and password.

The testing is performed in regards to some **risk** factors. The risk factors are:

- existing username: what happens if a user registers with an existing username?

- password too short: short password are a common security risk in a web application

Part I - Adding letters - Black Box Testing

Testing should be performed in an **analytical** manner, focusing on the factor of risk. Focus on the most vulnerable features first.

Again, testing is performed in regards to some **risk** factors. The risk factors are:

- extremely large data inputs: what happens if the users enters a very long address? is the letter accepeted?

- can you add letters with special characters or syntax, such as sql specific syntax?

**From Lecture 1B:**

There are 7 dimensions:

• scope, coverage, person who achieves testing, associated risks, activities, evaluation and desired results.

• All testing involves all dimensions.

• A testing technique focuses the attention on one or a few dimensions, leaving the others open to the tester’s judgment.

(From Lecture 1B)

# Test Design. Test implementation. Test execution. Test Report

# Test Design

*<Include all the information associated to the test design step when a specific test design technique is used. Each team member will fill out the details corresponding to Part I and Part II. The table below indicates a sample for Part Il.>*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Information objective (Part II): …* | | | | |
| **Student** | **Feature(s)** | **Test Design Technique** | **Details** | **Input, Expected output** |
| *Student1* | *Feature X* | *BT (Boundary Testing)* | *Variable n in [a, b] is tested* | |  |  | | --- | --- | | ***Input*** | ***Expected Output*** | | *a-1* | *…* | | *a* | *…* | | *a+1* |  | | *b-1* |  | | *b* |  | | *b+1* |  | |
| *Student4* | *Feature Z* | *LE  (Logical Expressions)* | *Business rules for variable a and b are tested* | |  |  |  | | --- | --- | --- | | ***Variable*** | ***Rule1*** | ***Rule2*** | | *a* | *a>0* | *a<=0* | | *b* | *b>0* | *b<=0* | | *result* | *False* | *True* | |  |  |  | |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Test Implementation. Test Execution

*<Each student will choose* ***ONE*** *of the test design techniques she/he included in the previous section. For the picked technique the test cases designed will be implemented using an automation framework (see* ***Lab04****) (SeleniumWebDriver + SerenityBDD, Postman, JMeter) should be emphasized. The team can decide if each student will have her/his own automation project, or they will share the same project on git such that it will include a package with the tests implemented by each team member. The table below will consist of the test cases implemented by each student.>*

|  |  |  |  |
| --- | --- | --- | --- |
| **Part** | **Student** | **Feature(s)** | **Input, Expected Output, Actual Output** |
| *Part II* | *Student1* | *Feature X* | |  |  |  |  | | --- | --- | --- | --- | | ***TCs*** | ***Input*** | ***Expected Output*** | ***Actual Output*** | | *TC01* | *a-1* | *…* | *… or* ***Passed*** | | *TC02* | *a* | *…* | *… or* ***Failed*** | | *TC03* | *a+1* |  |  | | *TC04* | *b-1* |  |  | | *TC05* | *b* |  |  | | *~~TC06~~* | *~~b+1~~* |  | *cannot be implemented* | |
| *Part I* | *Student4* | *Feature Z* | |  |  |  |  | | --- | --- | --- | --- | | ***TCs*** | ***Input*** | ***Expected Output*** | ***Actual Output*** | | *TC01* | *a=3, b=5* | *False* | ***False*** | | *TC02* | *a=-1, b=-3* | *True* | ***False*** | | *…* |  |  |  | |  |  |  |  | |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Test Report

*<This section will include the reports of the test execution, e.g., pie charts generated by the used tool, with #TCs run = #TCs passed + #TCs failed. If the entire team has worked on the same project, a single report/pie char should be provided. Otherwise, a pie chart/report should be presented for each team member.>*

# Issue Reporting

*<This section includes the application of the RIMGEA strategy for* ***at least one issue*** *found while performing testing. The type of issue can be coding bug or design issue. Highlight 2-3 relevant RIMGEA elements for the detected issue(s). Thereafter, report the bug or issue using the corresponding template (see* **IssueReport**). The bug/issues discussed in this section (and reported as well) refer to the entire team, not each team member.*>*

# Conclusions. Lessons Learned

*<Please include in this section final conclusions, lessons learned and personal considerations while working on TDTP (3-4 paragraphs). You can focus on the following aspects: type of application to be tested, amount of knowledge to use (related or not to testing), tools required to apply, team collaboration, test project organization, amount of time needed to fulfill the tasks, etc.>*