# Lab1. Testavimo veiklų planavimas

## Darbo tikslas

Sudaryti programinės įrangos testavimo planą.

## Pasiruošimas

Yra pasirinktas programinės įrangos projektas. Turimas ar bus sukurtas jo programinis kodas. Projektas gali būti kuriamas įvairiomis programavimo kalbomis naudojant įvairias technologijas.

## Informacija

Testavimo planas – dokumentas kuri sistematikai apibrėžia kaip bus testuojamas programinė įranga. Egzistuoja įvairūs testavimo planų šablonai. Daugumoje jie visi aprašo panašius dalykus. Vienas iš šablonų yra IEEE 829-2008. Pagrindinės testavimo plano dalys yra:

1. Testavimo apimtis – Apibrėžiama programinės įrangos apimtis, apibūdinama kas ketinama testuoti.
2. Testavimo strategijos – nurodoma kokios testavimo strategijos bus taikomos, pagrindžiama kodėl.
3. Pradinės sąlygos – nurodoma, kas turi būti atlikta, kad būtų galima pradėti vykdyti testavimo veiklas.
4. Testavimo prioritetai – sudaromas testavimo veiklų programinių komponentų testavimui prioritetų sąrašas.
5. Testavimo tikslai – apibrėžiami testavimo tikslai nurodant kokius sistemos aspektus svarbu patikrinti..
6. Testavimo technikos – nurodoma kokios testavimo technikos bus taikomos apibrėžtiems tikslams pasiekti.
7. Rolės ir atsakomybės – apibrėžiama testavimo procese dalyvaujanti komanda ir jos atsakomybės.
8. Rezultatai – kokie yra laukiami testavimo rezultatai, kaip jie bus dokumentuoti, pateikti.
9. Testavimo aplinka – nurodoma kokia programinė ir techninė įranga reikalinga testavimo veikloms atlikti. Kokiose įrangos konfigūracijose reikės išbandyti testuojamą programinę įrangą.
10. Testų scenarijai – apibrėžiami testų scenarijai, reikalingi išsikeltiems tikslams pasiekti ir sukurti pagal pasirinktas testavimo technikas.
11. Testų valdymas – apibrėžiama, kaip bus valdomi testai, kur registruojami. Apibrėžiama kur ir kaip bus registruojami defektai, kaip stebimas testavimo procesas.
12. Testavimo tvarkaraštis – pateikiamas testavimo veiklų atlikimo tvarkaraštis.
13. Testavimo rizikos – numatoma kas galėtu sutrikdyti testavimo veiklų vykdymą, apibrėžiama kaip minimizuojama rizika.

## Darbo užduotys

1. Pasirinkti programinę įrangą testavimui, trumpai aprašyti jos funkcionalumą (panaudos atvejai, funkcijų sąrašas, architektūra)
2. Iškeliami testavimo tikslai, nusprendžiama kokios technikos bus jiems pasiekti taikomis.
3. Apibrėžiamos testavimo aplinkos.
4. Užpildomas testavimo plano dokumentas atsižvelgiant į pasirinktus testavimo tikslus ir technikas.
5. Sukuriami/suplanuojami testų scenarijai.
6. Suplanuoti testų scenarijai patalpinami pasirinktoje testų valdymo sistemoje (pvz. qase.io) arba užrašomi testų užrašymo kalba (pvz. Gherkin naudojant Cucumber įrankį).

## Komentarai

1. Sekančiuose darbuose visų suplanuotu testavimo veiklų neprivaloma atlikti.
2. Sekančiuose darbuose vykdomo tik nedidelė dalis suplanuotų testavimo veiklų.
3. Planą galima pateikti lietuvių ar anglų kalbomis.

## Darbo gynimas

1. Sukurtas testavimo plano dokumentas.
2. Sukurti testai suvesti į testų valdymo įrankį (jei reikia, testų valdymo įrankį reikia įsidiegti).
3. Testavimo plano dokumentas įkeltas į Moodle.
4. Sukurti testai eksportuoti iš testų valdymo įrankio ir įkelti kaip dokumentas į Moodle).
5. Pasiruošta atsakyti į įvairius su darbu susijusius klausimus.

## Gynimui pasiruošimo klausimai

1. Kokiu tikslu kuriamas testavimo planas?
2. Kaip bus atliekami suplanuoti testų scenarijai?
3. Kada geriausia pradėti vykdyti suplanuotas testavimo veiklas?
4. Kodėl kažkurios testavimo veiklos buvo įtraukti į planą, kodėl nebuvo įtraukti?
5. Kodėl kažkurie testavimo tikslai buvo įtraukti į planą, kodėl nebuvo įtraukti?

## Nuorodos

1. IEEE 829-2008. IEEE Standard for Software and System Test Documentation.
2. <https://specflow.org/learn/gherkin/>
3. <https://qase.io/>

## Test plan: example/template

### Introduction

This document describes software test plan for orders processing system ABC.

### Tests scope

Acceptance testing consists of:

1. ABC web system, version V1.
2. Use cases models for customer, manager, administrator roles.

The purpose of this testing is to determine how well the implemented software matches functional requirements, also to identify the discovered problems and to allow to fix them. The testing also would allow collecting test data and test results for further software regression testing during its maintenance phase.

Users working habits could vary on user basic; this has to be disregarded during software testing.

### Test strategies

The basis of the acceptance testing is to demonstrate that software and its infrastructure are stable and performs reliably. The all-other testing activities must be completed before starting acceptance testing phase. The acceptance testing is oriented towards software testing from user perspective, in order to check how software is used in day-to-day scenarios and how it matches specified quality requirements. During the acceptance testing the representative from the customer Company will be present.

The additional testing activities will include unit testing, integration testing, security, performance testing.

### Prerequisites

Those tasks have to be completed before starting testing activities:

1. There exists complete software specification expressed in case and usage scenarios models,
2. Working software implementation,
3. Established procedure on fixing discovered issues during testing,
4. The set of defined use cases for acceptance testing for testing all software functionality,
5. The established testing environment,
6. Allocated testing resources,
7. Defined acceptance testing standards.

### Test priorities

The following testing activities are listed in decreasing priority level (the first has the highest priority):

1. Functions – do all defined software functions perform as expected?
2. Usability – is the software user friendly?
3. Security – are the data secured?
4. Performance – does the software matches agreed performance criteria?

### Test goals

1. Components testing (unit testing). The all software components are tested.
2. Integration testing. The software testing in order to ensure that components are interacting correctly.
3. Validation testing. Software testing in emulated production environment in order to test its functionality.
4. **Acceptance / Qualification Testing**
5. Software acceptance testing in order to ensure that its functionality matches end user expectations. Acceptance / qualification testing determines whether a system satisfies its acceptance criteria, usually by checking desired system behaviors against the customer’s requirements. The acceptance testing consists of the last set of tests that are executed before officially launching software system.
6. **Installation Testing**

Often, after completion of system and acceptance testing, the software is verified upon installation in the target environment

1. **Alpha and Beta Testing**

Before software is released, it is sometimes given to a small, selected group of potential users for trial use (alpha testing) and/or to a larger set of representative users.

1. **Reliability Achievement and Evaluation**

Testing improves reliability by identifying and correcting faults. In addition, statistical measures of reliability can be derived by randomly generating test cases according to the operational profile of the software.

1. **Regression Testing**

In practice, the approach is to show that software still passes previously passed tests in a test suite.

1. **Performance Testing**

Verifies that the software meets the specified performance requirements and assesses performance characteristics—for instance, capacity and response time.

1. **Security Testing**

Security testing verifies the confidentiality, integrity, and availability of the systems and its data. Negative testing.

1. **Stress Testing**

Stress testing exercises software at the maximum design load, as well as beyond it, with the goal of determining the behavioral limits, and to test defense mechanisms in critical systems.

1. **Back-to-Back Testing**

Testing in which two or more variants of a program are executed with the same inputs, the outputs are compared, and errors are analyzed in case of discrepancies.

1. **Recovery Testing**

Recovery testing is aimed at verifying software restart capabilities after a system crash or other “disaster.”

1. **Interface Testing**

Interface testing aims at verifying whether the components interface correctly to provide the correct exchange of data and control information.

1. **Configuration Testing**

In cases where software is built to serve different users, configuration testing verifies the software under different specified configurations.

1. **Usability and Human Computer Interaction Testing**

The main task of usability and human computer interaction testing is to evaluate how easy it is for end users to learn and to use the software.

### Test techniques

The following testing techniques will be used:

1. Tests scripts – the scripted uses cases (with predefined input and expected output data).
2. Test scripts without data – the testers will choose the input data during testing.
3. Unauthorized use tests – the scripted activities that try to gain access to the unauthorized data in the software.
4. Usability checklist – activities to evaluate systems the ease of use.
5. Performance statistics – performance characteristics collection and comparison with defined parameters.

### Tests management

There are defined the following roles and responsibilities:

1. Quality assurance lead – the person responsible for testing process planning and its execution.
2. Tester – performs testing activities defined in the test plan.
3. Product manager – ensures that tests are executed successfully from the user perspective.
4. Test support – ensures that technical equipment are in place and operational during testing.

### Results

After testing the following deliverables should be available:

1. Test plan – this document with all changes made during testing process.
2. Change requests – document describing software changes caused by changed requirements or discovered defects during testing.
3. Weekly testing progress reports.
4. Final sing-off document signed by customer confirming that system meats all functional and quality requirements.

### Testing environment

The following software and hardware configuration has to be available during software testing:

Server: One Intel based computer with the following configuration:

1. Xeon 4GHz, 16GB RAM, 10TB HDD, 1GB SSD,
2. Debian Linux 12.0,
3. Apache 2. Web Server,
4. Java 17
5. The Oracle database,
6. The working version of ABC software.

2 Workstations with the following configuration:

1. AMD 2GHz, 16GB RAM, 512GB HDD
2. Microsoft Windows 11 Professional
3. Safari 11 Browser

One workstation with the following configuration:

1. AMD 2GHz, 2GB RAM, 512GB HDD
2. Customers accounting system installed
3. Microsoft Windows 7 Professional

Additional hardware has to be present:

1. Installed testing network,
2. Server and 3 workstations connected to the local network.

ABS system configured in the following way, by adding the following users:

1. System administrator,
2. Client 1,
3. Client 2,
4. Manager 1,
5. Manager 2.

### Test scripts

#### Introduction

The following test scripts are destined for testing each software function. The part of those scripts reassembles the use cases scenarios. Those parts are not included into this document, those parts a linked instead.

Each tests script consists of:

1. Description – this part reassembles the description for usage scenario document.
2. Initial data – Usually its initial configuration of software database.
3. Test steps – the actions the tester has to perform during testing.
4. Test cases – tests input data and expected software reaction.

#### Test Script 01.0.1 Login (Fill into test managenet system or code in Gherkin, see appendixes A, B)

Description: see Tests login functionality.

Initial data: initial database.

Test steps: see usage Scenario.

Test cases:

|  |  |  |  |
| --- | --- | --- | --- |
| User | Password | Expected result | Comment |
| Administrator | 123 | Logged in |  |
| Administrator | Qweasd | Login failed |  |
| Manager1 | 123 | Logged in |  |
| Manager1 | Asdasd | Login Failed |  |
| Client1 | 123 | Logged in |  |
| Client1 | Asdasd | Login Failed |  |

#### Test script: 01.0.3. New order (Fill into test managenet system or code in Gherkin, see appendixes A, B)

Description: this tests test the new order placement function.

Initial data: initial data base, user has logged with client1/123 data.

Tests steps:

1. User navigates to new order page.
2. Fills order info.
3. Fills delivery address.
4. Confirms the order.

Tests cases:

|  |  |  |
| --- | --- | --- |
| Input data | Expected result | Comments |
| Submit type: Internet  Contact person: John Smith  Delivery: Yes  Delivery date: 2021.01.01  Delivery address: Kaunas Kamas 157  Quantity : 100 units  File to print: testimg.jpg | The order was accepted |  |

#### Test script 01.0.4. Review orders.

#### Test script 01.0.5. Monitor order progress.

#### Test script 01.0.6. Order payment.

#### Test script 01.0.7. Order canceling.

#### Test script 01.0.8. Edit order.

#### Test script 01.0.9. Submit order.

#### Test script 01.0.13. Cancel order.

#### Test script 01.0.15. Mark order for shipping.

#### Test script 01.0.16. Complete order.

### Tests management

Tests have to be managed adhering to company defined rules that include:

1. Tests execution.
2. Tests results reporting.
3. Defects monitoring and fixing.
4. Test environment configuration management.
5. Tests results configuration management.

### Testing schedule

|  |  |  |
| --- | --- | --- |
| Testing task | Start | Deadline |
| Unit testing | 2021-03-01 | 2021-07-12 |
| Integration testing | 2021-04-01 | 2021-10-23 |
| User interface testing | 2021-05-01 | 2021-11-22 |
| System testing | 2021-10-01 | 2022-01-10 |
| Acceptance testing | 2022-02-01 | 2022-05-20 |

### Testing risks

The possible risks that could influence testing process are:

|  |  |  |
| --- | --- | --- |
| Risk | Description | Mitigation |
| Not enough testers | Tester could be assigned to other projects | Ensure that this project testing activities have higher priorities |
| Customer managers could not be available during the testing process. | Client managers are always busy working on their business and have no time for answering tester questions or reviewing software. | Ensure that there is one manager allocated for our consultations on specified time once a week for 5 hours. |

### Appendix A. Test Cases in Qase system

<https://qase.io/>

Une image contenant texte

Description générée automatiquement

### Appendix B. Test Cases in Gherkin

Une image contenant texte

Description générée automatiquement