# Table of Content

[Table of Content 1](#_Toc68451248)

[Homework 1](#_Toc68451249)

[Test plan for Calculator of leasing 2](#_Toc68451250)

[**Test Strategy** 2](#_Toc68451251)

[**Current testing environment** 3](#_Toc68451252)

[**Entry and Exit criteria** 4](#_Toc68451253)

[**Features to be tested and not tested** 4](#_Toc68451254)

[**Testing shedule** 5](#_Toc68451255)

[**Additional material** 9](#_Toc68451256)

# Homework

Homework which contains automation as manual testing.

1. Get to know the application linked below (depending on country where you are applying):

·         EE: <http://www.seb.ee/eng/loan-and-leasing/leasing/car-leasing#calculator>

·         LV: <http://www.seb.lv/eng/loan-and-leasing/leasing/leasing-calculator>

·         LT: <https://www.seb.lt/eng/private/calculator-leasing>

 2. Define a prioritized test plan - how you would approach testing the application. Include test cases, priorities and select test cases that needs to be automated in your opinion. Test cases should be defined on a headline level (checklist), without detailed steps.  Also include your reasoning behind the approach (in a few sentences).

 3. Create automated test scripts either using your favorite Java test automation framework or Robot Framework/Python (choice is yours, select your favorite one) for 3 test cases chosen above.

 4. Present your test plan and link to repository with automated tests till **08.04.2021** in written form by email to [natalija.sakalinskiene@seb.lt](mailto:natalija.sakalinskiene@seb.lt)

# Test plan for Calculator of leasing

# Github: user: DGost123; password: Rimantas\_123;

## **Test Strategy**

Manual, semi-automated and automated testing is applicable. As test must be performed using Lithuanian and English user interface, data-driven testing automation framework is preferable.

Remark. Current test plan has been prepared and testing done using sequential automated testing framework (not data-driven).

Remark. Current test plan has been prepared for case when all testing is done by one experienced tester.

The following criteria should be taken into account when choosing the cases to automate:

* possible repeatability of tests according to test cases,
* probability of regression of software under testing,
* testing time consumption,
* test scripts’ development cost,
* type of testing needed.

Test cases for automated testing:

* negative and positive input data validation test cases (as there are a lot of various data and data combinations (that means repeatability));
* comparison of values calculated by “Calculator of leasing” with expected ones (as calculations may be performed using various data and their combinations (that means repeatability) - negative and positive test cases);
* display of data after “Add for comparison” button was pressed (including cyclic addition) ( as test scripts’ development cost is low and probability of regression exists);
* display of data after “Payment schedule” button was pressed (including schedule printing) – semi automated case (schedule printing is tested manually) (as test scripts’ development cost is low and probability of regression exists);
* display of data after “Schedule” link was pressed (including schedule printing) – semi automated case (schedule printing is tested manually) (as test scripts’ development cost is low and probability of regression exists);
* Smoke tests (to check whether the installed/deployed software build is stable).

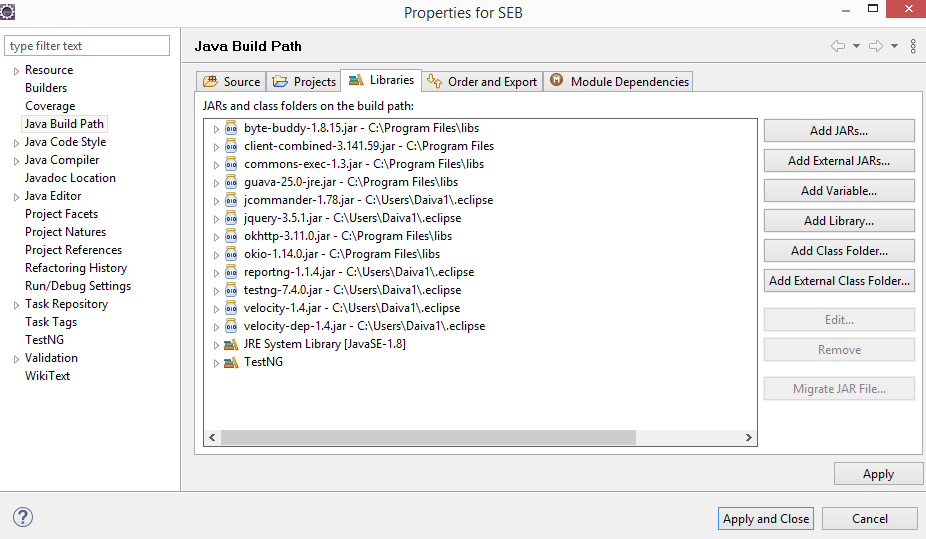
Remark 1: in case data-driven approach is not used, automated testing scripts prepared for testing the software which uses user interface in EN language should be modified in order to be appropriate for testing software which uses user interface in LIT language; different versions of scripts should be created for different languages.

Remark 2: if the logics of calculations is not complex, calculation of expected results should be programmed in the scripts for automated testing (otherwise programming costs would increase).

Remark 3: in order to minimize test scripts’ development cost, test cases for testing of software which uses EN and Lit user interfaces should be prepared in parallel.

## **Current testing environment**

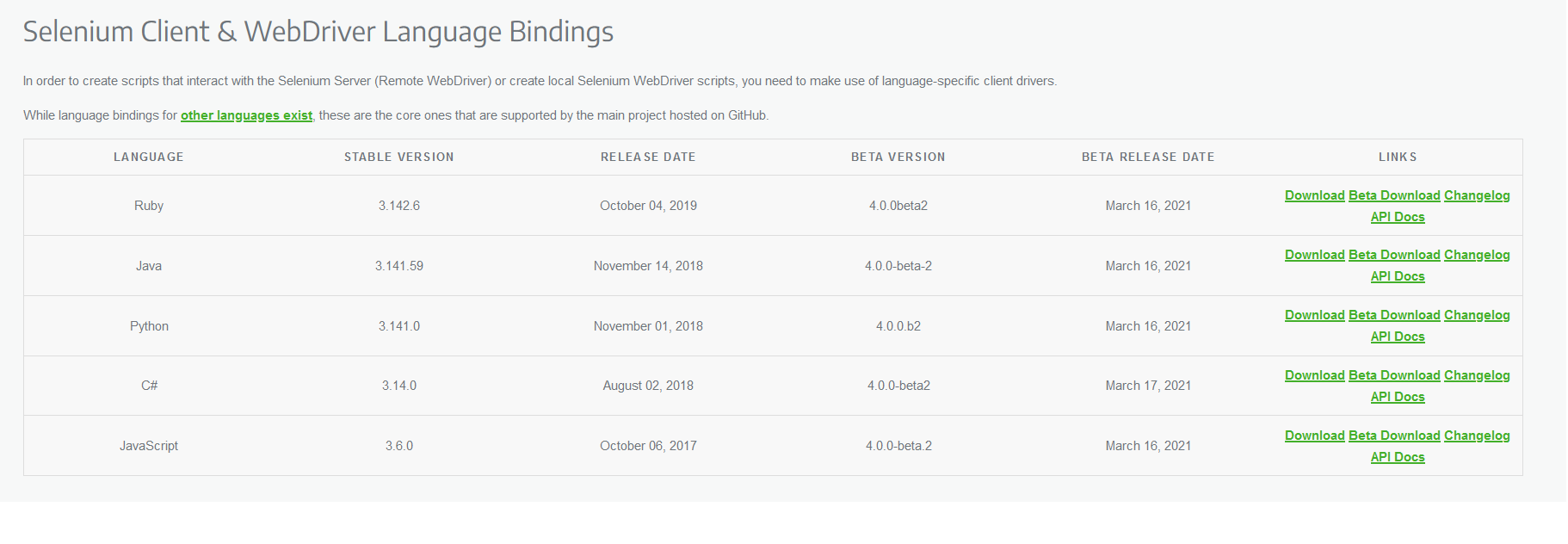
TestNG in Eclipse for Selenium WebDriver:

For execution of test cases and reporting of test results the following libraries must be installed:

How to Install Eclipse IDE on Windows 10: <https://www.youtube.com/watch?v=N-wXTRpR03U> ; <https://www.vogella.com/tutorials/Eclipse/article.html>; .

How To Install TestNG In Eclipse: <https://www.lambdatest.com/blog/how-to-install-testng-in-eclipse-step-by-step-guide/> ; <https://www.ecanarys.com/Blogs/ArticleID/169/How-to-Install-TestNG-framework-Step-by-Step-installation-process> .

Install Selenium Java client driver from : <https://www.selenium.dev/downloads/> :



## **Entry and Exit criteria**

Entry criteria:

* requirements specification approved and presented;
* approved test plan/strategy document;
* test environment setup is working as per the plan and checklist

Exit criteria:

* final test automation report prepared and approved;
* all important defects are logged and tracked to closure;
* all “nice to have” defects are logged.

## **Features to be tested and not tested**

To be tested:

* “Calculator of leasing” functionality testing;
* “Calculator of leasing” usability testing;
* SMOKE testing;

Not to be tested:

security testing;

stress, load testing;

## **Testing shedule**

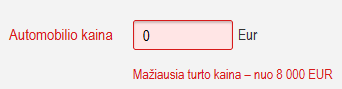
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Test cases** | **Priority (1- the highest)** | **Testing type** | **Method; remarks** |
| 1 | Static testing(verification of requirements’ specification) | 1 | Using check lists | Informal review |
| 2 | Test environment setup | 2 |  |  |
| Preparation of test cases for automated testing and automated testing (En) | | | | |
| 3 | test cases for the validation of values entered in the calculator’s fields (for negative and positive testing) (En) | 3 | Automated | Method: equivalence partitioning and boundary value analysis;  Cases:   * are values entered for mandatory fields; * are per cents <= 100; * are numeric values entered for purchase value, per cents, interest rate, First installment, remaining value; * are positive values entered for purchase value, interest rate, First installment, remaining value; * test case when purchase value <= 8000: For amount up to 8 000 EUR we offer [consumer loan](https://www.seb.lv/eng/loan-and-leasing/everyday-loans/consumer-loan): * check whether the link is present; * press the link and check whether it is valid and forwards to “Consumer loan“ page (?? Is this needed in case Lithuanian language is selected??) |
| 4 | test cases for comparison of values calculated by “Calculator of leasing” with expected ones (as calculations are made using various data and their combinations (that means repeatability) - negative and positive simpler cases (En) | 4 | Automated |  |
| 5 | test cases for display of data after “Add for comparison” button was pressed (including cyclic add for comparison) as test scripts’ development cost is low and probability of regression exist (En); | 5 | Automated, Smoke, regression |  |
| 6 | test cases for display of data after “Payment schedule” button was pressed (including schedule printing) – semi automated case (schedule printing is tested manually) as test scripts’ development cost is low and probability of regression exist (En); | 6 | Automated, Smoke, regression | * are values displayed in the table correct; * is new row added after each new comparison; * test for case when new values are entered and “Add for comparison” is pressed (“Calculate” button is and is not pressed before); |
| 7 | test cases for display of data after “Schedule” link was pressed (including schedule printing) – semi automated case (schedule printing is tested manually) as test scripts’ development cost is low and probability of regression exist(En); | 7 | Automated, Smoke, regression |  |
| Preparation of test cases for automated testing and automated testing (Lit) | | | | |
| 3a | test cases for the validation of values entered in the calculator’s fields (for negative and positive testing) (Lit) – only test cases that check the messages in Lithuanian | 3(for each test case: in parallel with 3) | Automated | Equivalence partitioning and boundary value analysis |
| 6a | test cases for display of data after “Payment schedule” button was pressed (including schedule printing) – semi automated case (schedule printing is tested manually) as test scripts’ development cost is low and probability of regression exist (Lt) - only test cases that check the messages in Lithuanian; | 6(for each test case: in parallel with 6) | Automated, Smoke, regression |  |
| 7a | test cases for display of data after “Schedule” link was pressed (including schedule printing) – semi automated case (schedule printing is tested manually) as test scripts’ development cost is low and probability of regression exist (Lt) - only test cases that check the messages in Lithuanian; | 7(for each test case: in parallel with 7) | Automated, Smoke, regression |  |
| Preparation of test cases for manual testing | | | | |
| 8 | test cases for comparison of values calculated by “Calculator of leasing” with expected ones (as calculations are made using various data and their combinations (that means repeatability) - negative and positive complex cases(En); | 8 | Manually |  |
| 9 | test cases for validation whether payment schedule is correct and formed according to the rules - complex cases(En); | 9 | Manually |  |
| 10 | test cases to check functionality of payment schedule printing: navigation to the next/previous/last/first pages; printing/cancelling of payment schedule report; checking the values and totals in payment schedule report; | 10 | Manually |  |
| 11 | test case when new values are entered and “Payment schedule” or link to Schedule is pressed (“Calculate” button is and is not pressed); | 11 | Manually |  |
| Manual testing | | | | |
| 12 | test cases for comparison of values calculated by “Calculator of leasing” with expected ones (as calculations are made using various data and their combinations (that means repeatability) - negative and positive complex cases(En); | 12 | Manually |  |
| 13 | test cases for validation whether payment schedule is correct and formed according to the rules - complex cases(En); | 13 | Manually |  |
| 14 | test cases to check functionality of payment schedule printing: navigation to the next/previous/last/first pages; printing/cancelling of payment schedule report; checking the values and totals in payment schedule report; | 14 | Manually |  |
| 15 | test case when new values are entered and “Payment schedule” or link to Schedule is pressed (“Calculate” button is and is not pressed); | 15 | Manually |  |
| 16 | Usability testing | 16 |  |  |
| 17 | Exploratory testing (scrolling, navigation, etc.) | 17 | Manual |  |
| 18 | Smoke testing | 18 | Manual, semi-automated, automated |  |
| 19 | Automated testing report preparation | 19 | Automated |  |
| 20 | Manual testing report preparation | 20 | Using defect management tools |  |
| 21 | Final testing report preparation | 21 | Automated |  |
| 22 | Improvements for future testing | 22 |  |  |

## **Additional material**

Remark: additional material is presented below. It lists the main test cases (out of scope of the task).

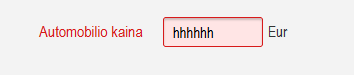
1. Test cases for the validation of values entered in the calculator’s fields (positive and negative cases):

* are values entered for mandatory fields;
* are per cents <= 100;
* are numeric values entered for purchase value, per cents, interest rate, First installment, remaining value;
* are positive values entered for purchase value, interest rate, First installment, remaining value;
* case when purchase value <= 8000:
  + - * + check whether the message “Mažiausia turto kaina – nuo 8 000 EUR” is displayed;











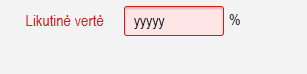








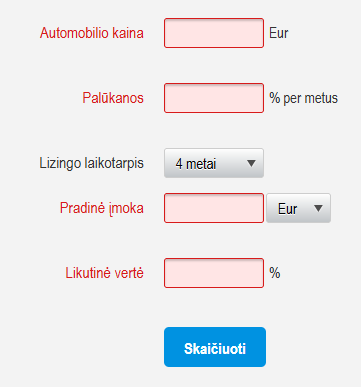




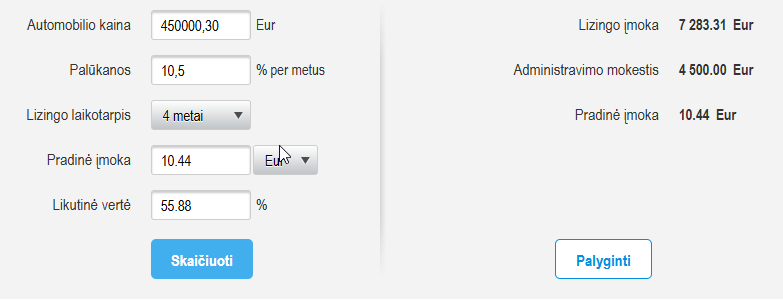


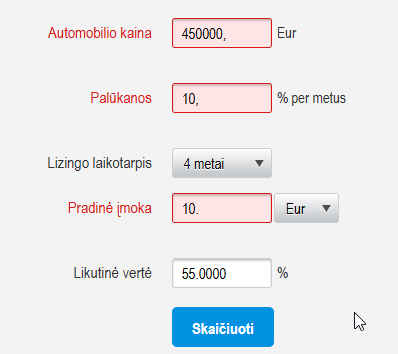


* test cases when values not entered:



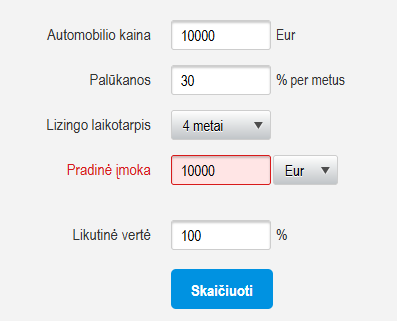
* When values entered are decimal:

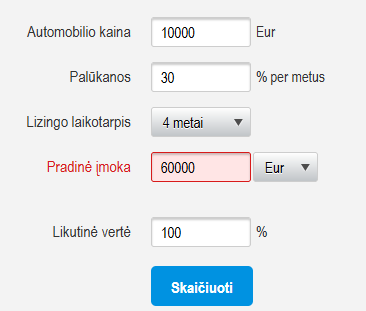






* Test case when pradinė įmoka>= automobilio kaina:





* Test cases for negative values validation:

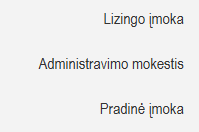




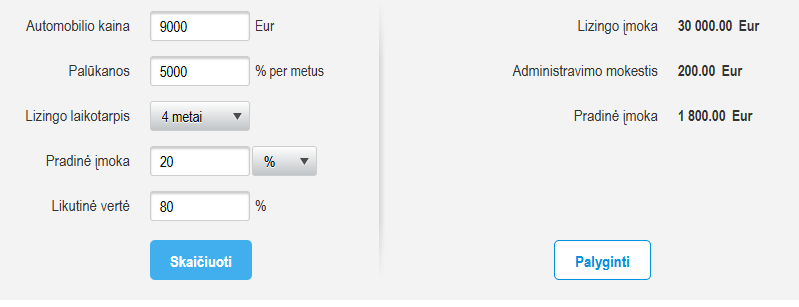




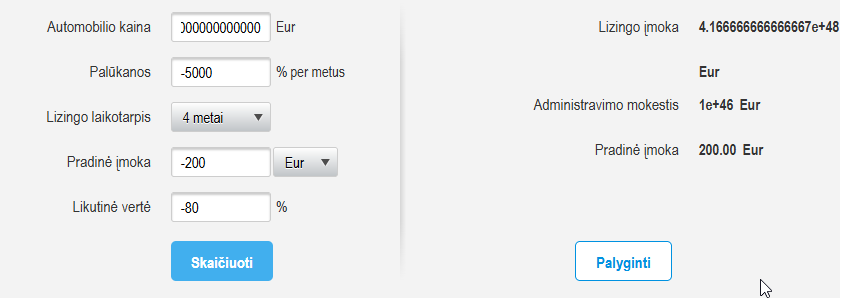
1. Test cases for the calculation logics (after pressing button “Skaičiuoti”), e. i. validate the resulting fields comparing them with expected values computed manually or otherwise, but not using Calculator of leasing:



Test cases for positive testing:

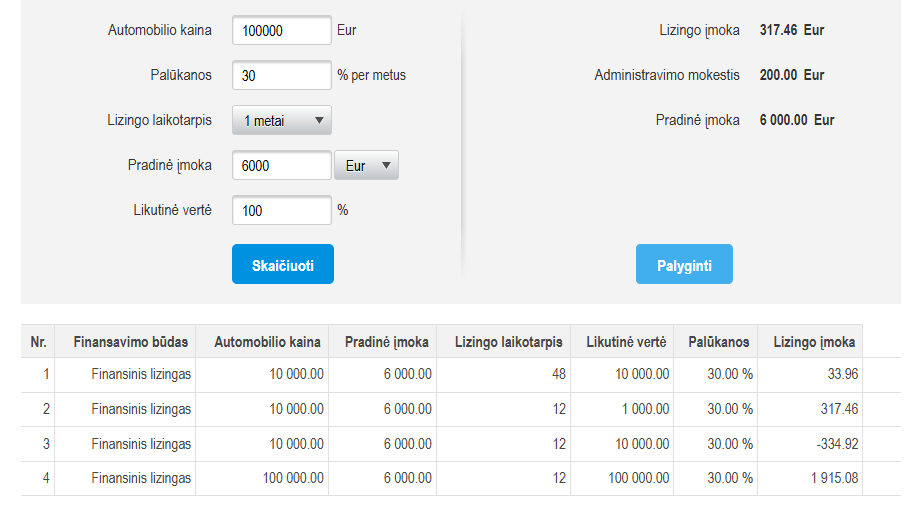


Test cases for negative testing:

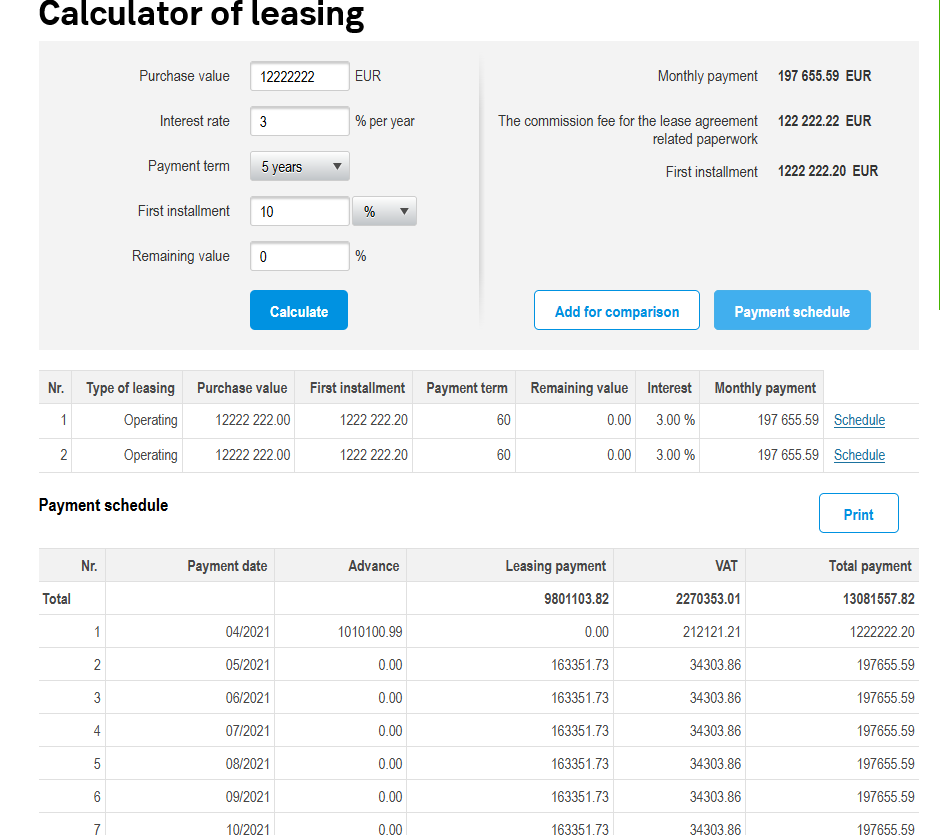


1. Test cases when pressing button “Palyginti”:

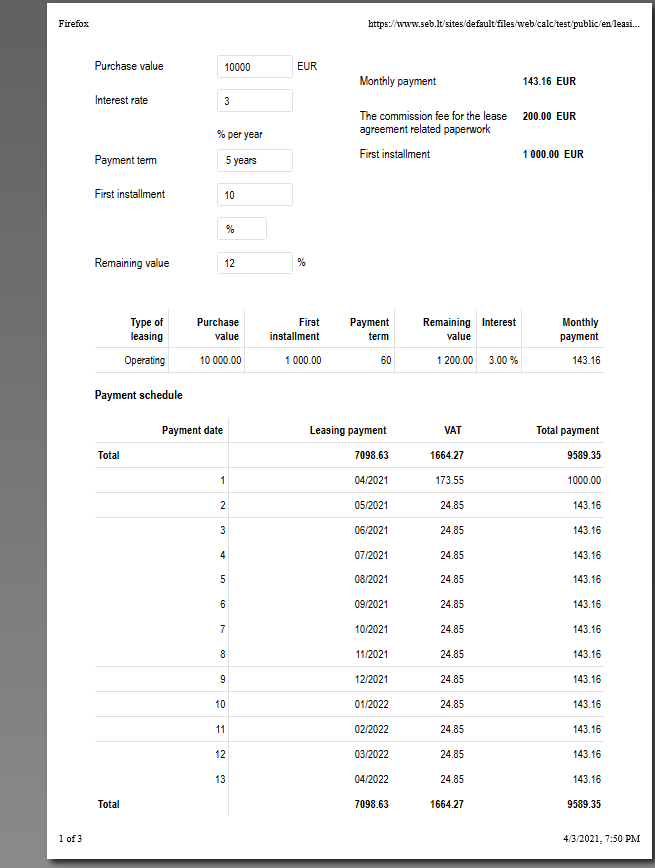
* are values displayed in the table correct;
* is new row added after each new comparison;
* test case when new values are entered and “Add for comparison” is pressed (“Calculate” button is and is not pressed before);



1. Test cases for Payment schedule



* test cases for validating whether payment schedule is correct and formed according to the rules- manual testing;
* test case for validating whether payment schedule “layout” is displayed for printing after “Print” button has been pressed;



* + test cases to check functionality of payment schedule printing: navigation to the next/previous/last/first pages; printing/cancelling of payment schedule report; check values and totals in payment schedule report – manual testing;
  + test for case when new values are entered and “Payment schedule” is pressed (“Calculate” button is and is not pressed);

Additional cases when English language is selected:

* test case when purchase value <= 8000: For amount up to 8 000 EUR we offer [consumer loan](https://www.seb.lv/eng/loan-and-leasing/everyday-loans/consumer-loan):
  + check whether the link is present;
  + press the link and check whether it is valid and forwards to “Consumer loan“ page (?? Is this needed in case Lithuanian language is selected??);
* test case for displaying payment schedule in En when button “Payment schedule” is pressed;