Project Plan for Degree Projects
Department of Computer Science

General information

Title:	Reduction of test cases by utilizing the GUI in regression testing
External company:	Svenskaspel AB

Persons involved

Student 1:	Christer Hamberg	ch222xb@student.lnu.se
Student 2:		

Supervisor:	Johan Hagelbäck
External supervisor:	Michael Olofsson

Background

Running true end to end regression testing of a web solution is not easy. The main area of concern is the GUI (the application running in the web browser). One of the tools used to drive the navigation automatically in a browser is Selenium WebDriver, which will be used in the implementation part of this thesis.

Challenges are related to:

- The execution time is longer using a web browser (GUI) compared to only sending API requests to the backend (commonly implementing the business logic).
- Flaky test results due to browsers getting stuck or components not being updated.
- Keeping tests running due to modifications of the GUI components, which impacts the navigation in the test cases.

The common solution to these challenges is to separate the testing of the GUI (look, feel and navigation), with testing of the APIs (business logic). And then use a small set of test cases that covers the true end to end scope.

The result of this approach is however a bigger set of test cases in total due to partly overlapping test cases, were the full functionality is not tested in one and the same case. Less coverage of end to end testing of the business requirement.

If the above mention challenges can be addressed, it will be possible to reduce the total amount of test cases needed to regression test the business requirements, as well as achieving better test coverage of the requirement.

Problem formulation

The time gained by testing the parts separately also comes with a cost, more test cases in total needs to be maintained. End to end test coverage is smaller than it could be, thus opening up for possible interoperability faults between the GUI (frontend) and the Backend implementing the business logic.

By studying the factors limiting the use of the GUI in a true end to end commercially available web solution and addressing these issues the intention is to show that the GUI can be part of the regression test.

There are known limitations that the law of physics already imposes on this theses. It is not expected to be able to show that running test cases in a GUI can ever be as quick as only sending the required API calls to test the same functionality. However I do think that it is possible to address the issues causing slow interaction with the GUI so that good enough execution time can be achieved. I also think that it is possible to achieve a stable and reliable test suite for regression testing also when the GUI is included in a commercially available web solution.

This research would benefit any company interested in gaining a better end to end test coverage of their web solutions. As well as reducing the amount of needed test cases in their regression test by incorporating the GUI in the test cases.

Time to be spent on the thesis is limiting the scope of the research to a subset of the betting products provided by Svenskaspel.

Motivation

Testing of software is continuously changing over time. A current trend is Continuous Integration, Continuous Delivery and Deployment. This means that the software being developed should always be deployable and features can be deployed whenever.

This puts a restraint on regression testing when not only speed is an issue but also the scope covered. This thesis addresses one of the most problematic areas in regression testing, how to include the GUI in end to end regression testing, and limit the effects of the pitfalls by using the real GUI when performing regression testing.

Research Questions

RQ1.	What are the top 2 issues using Selenium WebDriver that cause longer test case execution times when test are run using a browser, compared to testing the same functionality with APIs, and what can be done to minimize the time differences?
RQ2.	What are the top 2 issues using Selenium WebDriver causing flaky (unreliably) test results, and how can these be overcome?
RQ3.	What are the top 2 issues using Selenium WebDriver causing test cases to break when GUI modifications are done, and how can these be overcome?

Method

The research method used is inductive research, where knowledge will be derived from the observation of experimental tests.

The research questions addresses both time and quality aspects. In order to be able to address the research questions experimental tests will be conducted on a few of Svenskaspels's betting games Keno, Vikinglotto and Eurojackpot.

1. An experimental test suite addressing both testing with APIs and Selenium WebDriver (GUI navigation) will be developed. This gives a base to compare the two test approaches.
2. The test suite will be run against the Web solution and data regarding execution times and type of failures will be collected. To be able to generate a good result the same test suite will be run several times before the collected data is analyzed.
3. Analysis of the collected data
4. Experimental changes on the test suite will be implemented after the analysis and the test suite will be run again (several times) against the same Web solution.
5. The result of the modifications will be analyzed and compared with the original result in order to see if they improve the addressed issue/problem or not.
6. The Experimental changes will be repeated until the result, data collected is considered to be good enough.

A single test of the suite is not considered to be good enough, hence the same tests has to be repeated over and over again. Also it would not be possible to address the issue with flaky test results with only one test run.

By repeating the test runs, implementing modifications and reevaluating these we aim to achieve a good enough working solution where the research questions have been answered and appropriate measures have been tested to address the issues causing long execution times, flaky test results as well as keeping test cases running.

Time plan

Week	Task	Deadline
2	Project Description	23.1.2017
5	Project Plan	13.2.2017
7	Literature study, Implementation (Draft 1 ready)	5.3.2017
8	Implementation done, game1 (data collection started) Peer review 1	12.3.2017
9	Implementation done, game2 (data collection started) (Draft 2 ready)	19.3.2017
10	Implementation done, game3 (data collection started) Peer review 2	26.3.2017
11	Implementation done, game4 (data collection started)	2.4.2017
13		
14	Peer review 3	23.4.2017
15		
16	Peer review 4	7.5.2017
17	Peer review 5	14.5.2017
18		
19	Final report, PPT for the Opposition report	28.5.2017
20	Final presentation, Opposition report	

The documentation part follows general timetable which can been found here: https://coursepress.lnu.se/kurs/examensarbete-kandidatniva/tidsplan/

However the implementation needs to be completed earlier in order to get good enough quantitative data collected over time. Hence more time than "available" will be spent the first few weeks in order to get the implementation made as soon as possible.

References