Test task for Senior Quality Assurance Engineer

05. Unit SQA

Exported on 03/07/2019

Table of Contents

1 Overview

This document describes the standard test task that is used to evaluate the skills and framework knowledge of applicants for the Automation QA Engineer position.

2 Goal

The goal of the task is to evaluate a number of criteria that are important for an Senior Quality Assurance Engineer's daily job performance:

- Ability to understand business requirements.
- Ability to design meaningful test cases based on business requirements and provided solution.
- Ability to design an automated testing solution for those test cases.
- Executing test cases.
- Understanding of and proficiency in the tools and technologies that are commonly used in Westernacher's solutions.
- Coding style.
- Eye for detail, correctness, ease of use.
- Documentation of test results and bug reports.
- Ability to cut down software problems in small, reproducible steps that can be easily understood by the engineering team.

3 Task description

3.1 Your activities

- Planning and Controlling
  - provide a short WBS (Work Breakdown Structure) for the task
  - provide a short effort estimation
- Analysing and Designing
  - Design test case specification derived from the requirements below and document them
- Realisation and Execution
  - Realise the test case
  - automate the test case execution
  - execute them
- Evaluating and Report
  - Evaluate the execution and compile a small report
- Closure
  - <no tasks>

3.2 Test conditions

3.2.1 Business requirements

The development team is building a small web application that manages employees' leaves in a database. The application has the following main functionalities:

- A list page that shows all current user's leaves in a sortable grid and a calendar component that shows his and his teammates leaves.
- Create new leave dialog that adds new leave to the database.
- Ability to edit a leave.
- Ability to request a leave.
- Ability to export a leave document, after the leave is requested
- Ability to delete a leave
- Multi-language support
- Account details (optional module to be tested)

3.2.2 Implemented Solution

The development team has delivered a solution to the QA team:

https://test.easyleave.de/

The login credentials will be emailed to you separately by the recruitment manager.

Your task is to create a suite of test cases, which you should later execute. These test cases should **not** only cover the happy paths, so use your imagination. As the provided solution has even more nice to have features, due to the dev team really wanted a well-done and easy to use application, you should also create test cases for these and execute them. As the solution will be delivered to the client as a first iteration the Product owner would like all functionalities to be covered.

You should also create a suite of automated tests that test the application. Also a report should be created for the PO with found issues and proper description of the steps to reproduce.

- Functional tests are mandatory
- REST API tests are optional but very nice to have
- Penetration tests are optional but very nice to have
- Performance tests are optional but very nice to have
- Other levels and areas of testing, which the applicant finds necessary

The tests should execute successfully on the provided url. Required browser is Google Chrome, other browsers are optional but very nice to have. We'll leave the design of the test suit up to you.

3.3 Test framework conditions

3.3.1 Architecture

The test suite should use an architecture that's similar to the way similar test suites are built at Westernacher. This means:

- Java 8+
- jUnit
- Selenium Web Driver
- Selenium Standalone Server, preferably setup in Grid mode
- (optional) Selenium IDE
- (optional) RESTful calls to the solution's backend
- Alternative approach - Groovy and Geb

3.3.2 Build

The build tool used should be Maven or Gradle (preferred).
Make sure to add build tasks that make it easy to build, deploy and run the test suite from scratch. Assume we are using continuous integration server and we would like these tests to be executed before every deploy of the application.

3.3.3 System environment

The environment components should come from this list:

- Selenium Standalone Server 2.45 or higher
- Google Chrome, latest version
- Java 8+
- Windows 7/8/10, Mac OS X 10.9 or higher, Ubuntu LTS 12.04 or higher

3.4 Documentation

- Please provide all required documentation (estimations, testcases, reports, etc.) in a common format (e.g. doc, xls, csv)
- Please provide a short documentation that explains how we can build and deploy the test suite. Make sure you list the system prerequisites.

- We don't need end user or code documentation, we'll figure that out ourselves.

3.5 Final thoughts

- Try to make sure that your application is an example of a state-of-the-art testing suite so you should convince us that you're the right person to join our team.
- Try to keep the codebase as compact as possible.
- Try to program defensively.
- Be creative. If you think you found something nice you want in the application to show us your skills - go ahead 🙂
- Be focused. You don't have to fulfill all objectives. You should balance between the time you spend and the degree of enabling us to understand your way of working. The only mandatory objective will be to convince us that you are the right candidate.

4 Reward

We don't want you to create this test application for free! If your work is approved by our reviewers and you successfully fulfil your trial period at Westernacher, we'll pay you **a bonus of 500 LEV as a compensation for your efforts.**