SOFTWARE ENGINEERING

SOFTWARE TESTING

|  |  |
| --- | --- |
| TABLE OF CONTENTS |  |
| **EXECUTIVE SUMMARY** | **2** |
| **SOFTWARE VALIDATION AND WHY IS IT IMPORTANT?** | **3** |
| **TEST-DRIVEN DEVELOPMENT** | **4** |
| **IS THIS APPROACH TO SOFTWARE VALIDATION COMPREHENSIVE?** | **7** |
| **APPLICATION OF DIFFERENT APPROACHES TO SOFTWARE TESTING** | **7** |
| **CONCLUSION** | **8** |
| **BIBLIOGRAPHY** | **9** |
| TABLE OF FIGURES |  |
| FIGURE 1 | 5 |

# EXECUTIVE SUMMARY

OctoFlex Technologies is a start-up IT firm that is focused on fulfilling all the needs of the businesses. We provide services to small- and medium-sized companies. With experienced professionals and a team approach to most of the projects, OctoFlex Technologies will be able to offer a more balanced quality services than many of its competitors in the market.

## The Company

OctoFlex Technologies is a team of two IT professional. Each professional specializes in some particular discipline and has broad knowledge of other disciplines as well. OctoFlex Technologies offers many services for business owners to select from, depending on their individual needs. This includes developing a website or application on any platform for any buisness.

## The Market

OctoFlex Technologies promises to provide the best service and customer experience. Its main goal is to fulfill all the needs of its customers. OctoFlex Technologies assures that the final product that it will develop will be of good value.

## Description On Software Testing

Testing is something that intends to show what a program or an application is intended to do and it shows what are the defects before the whole program or application is put into use. Testing is a part of more general verification and validation process. Testing includes both validation and verification. This report outlines the way we will be validating the software that we will develop. The approach that we will take to validate the software delivered to the client. We will also provide detailed description about the tools and procedures used to test the software. We will discuss about the reason why this approach is comprehensive. Some applications of the other different approaches to test the software will also be described.

# SOFTWARE VALIDATION AND WHY IS IT IMPORTANT?

Software Validation is the process of verifying and reconfirming what the design of the software is for and whether it satisfies all its intended purpose. Software Validation involves installation, procedures, testing, and taking reviews from the client and the other end-users. Validation with verification are the aspects of quality assurance and quality control. Validation is considered as ‘Are we building the right software’ and verification is considered as ’Are we building the software right’. For software verification, the software should comply with its specification while in software validation, the software should do what the user want it to do. Software Validation is a challenge. For example, we do not know how much validation is required for a particular software. We do not know its scope. As there will be many potential users, and potential features in the software, it will be difficult to manage the software.

There are many reasons why software validation is advantageous:

* Test cases are performed in a step-by-step process to make sure the software correctly selected and deployed.
* Large set of records are made which helps in resolving diagnostic and troubleshooting issues in the future.
* It provides alignment between all the stakeholders of the software.
* Software Validation provides some assurance that the implementation of software upgrades and changes does not affect the software quality.

Software Validation is required in many in many industries. OctoFlex Technologies Inc. will be using a unique approach to software validation that has some procedures and tools to test the software for different type of industries.

For example, for aerospace or car manufacturing and repairing, the goal will be to check if the software is reliable and safe to use on. Software systems in the laboratories used in research or testing medical devices should be validated to make sure that all the records are up to date, complete and accurate. The project that OctoFlex Technologies Inc. is working on is Blood Support+. For instance, it also requires up-to-date and complete records. The validation is required by law in some cases as there are some chances of software to have some quality issues which might endanger someone’s safety. Software Validation is beneficial for the industries with general purpose because there are chances of having quality issues in the software that they use for calculations.

OctoFlex Technologies Inc. will be using a Test- Driven Development Approach to validate the software that it delivers to its clients. The details on the software test-driven development approach are as follows:

# TEST-DRIVEN DEVELOPMENT

Test-Driven Development or TDD is a software development methodology in which test cases are written to draft and assess the way the code will perform. It is an approach to program development in which we inter-leave testing and code development. The Agile Manifesto and Extreme Programming are the ancestors of the TDD methodology. The test process, as the name suggests, directs software development. Furthermore, it's a structuring technique that enables programmers and testers to get optimized code which turns out to be durable over time.

There are several phases a software goes through in TDD which are:

* Identifying the new functionality: We first start by identifying the increment of functionality that is required and is important. These incremental functionalities should be normally small and should be able to implement in few lines of codes.
* Write precise tests: To confirm the functionality of features, we must write precise unit tests. For the test to run, they must make sure that it compiles. The test will typically fail. We are building compact tests based on their predictions of how the feature will operate, so this failure has significant implications.
* Run Tests (Correcting the Code): Once a test fails, we will implement the functionality and rerun the test. We need to make the minimal changes required to correct the code so that it can run successfully when re-executed.
* Refactor the Code: Once the test runs successfully, check for redundancy or any possible code optimizations to enhance overall performance. Ensure that refactoring does not affect the external behavior of the program. Then we move on to the next chunk of functionality.



Identify new functionality

pass

Write test

Run test

fail

Implement functionality and refactor

**FIGURE 1**

As stated above, writing unit tests is important for writing precise tests. We will use Unit testing as an automated procedure that ensures that even small code components are working correctly. Unit tests are done individually. Individual software units or components are tested as part of unit testing, a type of software testing. The goal is to confirm that every piece of software code functions as intended. The developers test the code of an application while it is still in the development (coding) stage. Unit tests isolate a piece of code and ensure its accuracy. A unit could be a single action, method, step, module, or object.

There are some benefits of Unit Testing which are:

* Early problem detection is possible.
* Without affecting other modules, changes are possible.
* Module integration is made simpler.
* Makes Design and Documentation easy.
* Decreases the number of bugs and the amount of time needed.

Although Unit Testing is useful, there are some challenges to perform it. Some of them are listed below:

* Problems with test names.
* Wrong test types being written.
* It is tedious to comprehend the entire code.
* A doubles test is necessary.
* Incorrect initial circumstances.
* Identifying dependencies.

So, it is important to make sure that the unit test cases are well outlined.

The tools (or frameworks) that we will be using with Test-Driven Development or TDD to run our program tests for the software development are JUnit and Jest. We will use both JUnit an Jest for validating front-end and back-end of our software application.

JUnit is a Regression Testing Framework that we will use to implement unit testing in Java, speed up programming, and improve code quality. The JUnit Framework is simple to integrate with other programs. Whereas Jest is a JavaScript library for writing, executing, and structuring tests. It is a test runner for JavaScript.

The main reasons to take up JUnit Testing are:

* It boosts the confidence of the developer and motivates them immensely.
* Early bug detection, which boosts the dependability of the code, during the development phase.
* The framework enables the developer to spend more time reading code than writing it. This enhances the readability, dependability, and bug-free nature of the code.

The main reason to take Jest for testing is because in many cases unit tests like Junit will not provide accurate results or function well when they will be performed on the front end of the software. Jest can validate almost anything in JavaScript. Jest can be used to validate the web application codes rendered on browser. Jest has become widely used JavaScript testing framework nowadays because of its easy setup and installation.

# IS THIS APPROACH TO SOFTWARE VALIDATION COMPREHENSIVE?

Yes, this approach to Software Validation is Comprehensive because it outlines the overall plan and approach to validating the software. This includes the scope of the validation process, the schedule of all the validation activities to take place, and most importantly it outlines the first to last, foremost important to least important features will be created.

# APPLICATION OF DIFFERENT APPROACHES TO SOFTWARE TESTING

There are some other different software testing processes that are commonly used to validate a software which are:

* Unit Test: We have already stated about it above. Verification of single classes are done in unit test.
* Module Test: Testing of interaction between group classes or packages takes place in the module test.
* Integration Test: Verification of interaction between modules or small components are done.
* System Test: Testing the system against its objective or goal is done in system testing. Release Testing is a part of System Testing which is done to convince the supplier of the system that the software is good enough for usage. Performance Testing is a subdivision of system testing and a part of the release testing to check the performance and reliability.
* Acceptance Test: Validation of the software application is done here to check if the software meets all the user requirements so that it can be acceptable.
* Regression Test: Re-running and refactoring all the tests on the system or software when it has changed.

# CONCLUSION

Overall, Software Validation is the process of verifying and validating the design of the software and whether it fulfills its desired purpose. Software Validation has its own advantages and is used in different type of industries to check the devices or automated technologies against their objective. Test-Driven Development Approach to Software Validation is widely used and is very comprehensive. It goes through several phases such as identifying new functionality, writing precise code, run tests, and refactoring the code. It has some advantageous which overrule its some of its disadvantages if the tests are well outlined. The tools (or frameworks) best suitable for us to do software validation would be JUnit and Jest with TDD. There are several other approaches to software testing which are: Unit Test, Module Test, Integration Test, System Test, Acceptance Test and Regression Test.

# BIBLIOGRAPHY

*20 most popular unit testing tools in 2023*. Software Testing Help. (2023, January 11). Retrieved January 31, 2023, from https://[www.softwaretestinghelp.com/unit-testing-tools/](http://www.softwaretestinghelp.com/unit-testing-tools/)

Gaba, I. (2022, November 24). *What is junit: An overview of the Best Java Testing Framework*.

Simplilearn.com. Retrieved January 31, 2023, from https://[www.simplilearn.com/tutorials/java-tutorial/what-is-junit](http://www.simplilearn.com/tutorials/java-tutorial/what-is-junit)

Hamilton, T. (2022, December 31). *Unit testing tutorial – what is, Types & Test example*.

Guru99. Retrieved January 31, 2023, from https://[www.guru99.com/unit-testing-](http://www.guru99.com/unit-testing-) guide.html

*Jest Tutorial - JavaScript unit testing using JEST framework*. Software Testing Help. (2023, January 22). Retrieved January 31, 2023, from https://[www.softwaretestinghelp.com/jest-](http://www.softwaretestinghelp.com/jest-) testing-tutorial/

Rony, H. (n.d.). *Software testing approaches and Techniques*. C# Corner. Retrieved January 31, 2023, from https://[www.c-sharpcorner.com/article/software-testing-approaches-and-](http://www.c-sharpcorner.com/article/software-testing-approaches-and-) techniques/

*Test Driven Development (TDD) : Approach & Benefits*. BrowserStack. (2021, January 15). Retrieved January 31, 2023, from https://[www.browserstack.com/guide/what-is-test-](http://www.browserstack.com/guide/what-is-test-) driven-development

Twona. (2023, January 27). *Is software validation outdated?* TwonaBlog. Retrieved January 31, 2023, from https://[www.twonas.com/blog/is-software-validation-outdated/](http://www.twonas.com/blog/is-software-validation-outdated/)

*Unit testing: A detailed guide*. BrowserStack. (2022, December 28). Retrieved January 31, 2023, from https://[www.browserstack.com/guide/unit-testing-a-detailed-guide](http://www.browserstack.com/guide/unit-testing-a-detailed-guide)

*What is jest – a tutorial on how to use jest*. What Is Jest – A Tutorial on How to Use Jest. (n.d.).

Retrieved January 31, 2023, from https://[www.lambdatest.com/jest](http://www.lambdatest.com/jest)