Test plan

Windesheim

Spark! Living Lab Conditioned Goods

Version 0.8



**Authors:**

|  |  |  |
| --- | --- | --- |
| Name | Student number | Email |
| Tim Imming | S1132262 | S1132262@student.windesheim.nl |
| Thijs Mansveld | S1127391 | S1127391@student.windesheim.nl |
| Sjoerd van de Kerkhof | S1129492 | S1129492@student.windesheim.nl |
| Gerard Wesseling | S1131366 | S1131366@student.windesheim.nl |
| Berat Guzel | S1127994 | S1127994@student.windesheim.nl |

**Version control**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Who | What |
| V0.1 | 2-6-2021 | Sjoerd van de Kerkhof, Gerard Wesseling | Introduction, Test strategy, Results |
| V0.8 | 2-6-2021 | Sjoerd van de Kerkhof, Gerard Wesseling | Finalized |

**Distribution**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | When | Addressee | Reason |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Approval**

|  |  |  |
| --- | --- | --- |
| Signature | Date | Name |
|  | 4/06/2021 | Maxime Bouillon |

Index

[1 Introduction 3](#_Toc73542334)

[2 Test strategy 4](#_Toc73542335)

[2.1 Unit tests 4](#_Toc73542336)

[2.2 Integration tests 4](#_Toc73542337)

[3 Results 5](#_Toc73542338)

[4 References 6](#_Toc73542339)

[5 Appendix 7](#_Toc73542340)

# Introduction

A cold chain is a temperature-controlled supply chain, where the cargo is cooled throughout the chain. These cold chains are arranged traditionally. There is minimal insight into the delivery times, conditions, and origin of the goods within the chain. Lamb Weston, a major producer of French fries, uses a cold chain to transport their fries from the factory to a cooled warehouse. They want to measure the temperature during the transport from their factory to the cold store to guarantee that the fries are cooled correctly, and the conditions are as stated by the provider for the cold chain’s entirety. The sensor data needs to be stored without any entity being able to alter the information.

A previous project group advised Spark! Living Lab that a blockchain, specifically Hyperledger Fabric, could provide these functionalities. Because of this advice Spark! Living Lab commissioned this project. The goal of this project was to build a Proof of Concept to show if Hyperledger Fabric meets the requirements for this use case.

This document describes the test strategies that were used to test different aspect of the Proof of Concept. It will only contain tests for the Hyperledger Fabric applications, because the Hyperledger Fabric’s infrastructure was not complete at the time of writing.

# Test strategy

The applications will be tested to check if everything functions as it was designed. The applications will be tested in different ways. In this chapter will be described how the team tested the different applications.

## Unit tests

Unit tests are automated tests to test the internal working of the business logic. The unit test will test the internal business logic by performing multiple different scenarios and checking the responses. The tests are designed to be independent from each other because the test can be performed randomly in different order every time they run. Unit tests are used to test the business logic of the smart contracts defined in the chaincode. To implement this the team uses Jest, a JavaScript testing framework based on Jasmine.

## Integration tests

Integration tests are automated tests to test if the application works as designed by using the application the way it should work. These tests will be performed with multiple different scenarios and the response will be checked. The tests are also designed independent from each other because the test can be performed randomly in different order every time they run. Integration tests are used to test the workings of the Application Programming Interface (API). Because the API is just a communication layer between the blockchain and third parties it does not have much internal business logic. This makes it not beneficial to use unit tests here. To implement this the team uses a combination of Jest, a JavaScript testing framework based on jasmine, and Supertest, a library to test HTTP servers.

## Performance test

Performance tests are automated tests to test the performance of the blockchain network with a large number of requests.

# Results

This chapter contains the results of the previously mentioned tests.

## API Routes

Text

Description automatically generated

Figure 3‑1: API Route integration test result

Text

Description automatically generated

Figure 3‑2: API Route integration test result

As shown by figure 3-1 and 3-2 all integration tests were successful. The first tests showed some small inconsistenties in the chaincode and API. The inconsistenties have been rectified.

## Chain code

Text

Description automatically generated

Figure 3‑3: Chaincode unit test result

Text

Description automatically generated

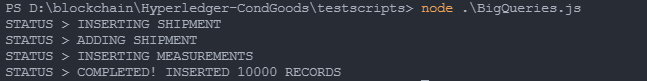
Figure 3‑4: Chaincode unit test result

As shown by figure 3-3 and 3-4 all unit tests were successful. The first tests showed some small inconsistenties in the chaincode and API. The inconsistenties have been rectified.

## Performance test



The performance test showed that the blockchain infrastructure could handle at least 10 000 requests as partly shown by figure 3-5.



# References

This document contains no references.

# Appendix

This document contains no annexes.