

Quality ICT B.V.

Client

Functional Design

Christopher Sulistiyo (4850025)

Christopher.sulistiyo@student.nhlstenden.com

ICT & IC Information Technology Department Emmen

Version 1.1 – 21/02/2024

[Version control 2](#_Toc158967910)

[Remarks 2](#_Toc158967911)

[Introduction 3](#_Toc158967912)

[Reasons for the Research 3](#_Toc158967913)

[Research Objectives 4](#_Toc158967914)

[Research Main and Sub-Questions 5](#_Toc158967915)

[Concept Theory 6](#_Toc158967916)

[Concept clarification 6](#_Toc158967917)

[Definitions to apply to the knowledge base. 7](#_Toc158967918)

[Sources validation within the company or employees 11](#_Toc158967919)

[Research Setup 12](#_Toc158967920)

[Design/ Data Collection 12](#_Toc158967921)

[Population and Sample 12](#_Toc158967922)

[Analysis Proposal 13](#_Toc158967923)

[Internal & External Validity and Reliability 13](#_Toc158967924)

[Usability 13](#_Toc158967925)

[Research Tasks 15](#_Toc158967926)

# Version control

|  |  |  |
| --- | --- | --- |
| **Version** | **Activities** | **Date** |
| Initial version 1.0 | Draft version | 06/02/2024 |

Remarks

Any changes and new developments that have a significant impact on the project proceedings will be noted here.

# Introduction

## Background and Context

This document is meant to provide an overview of the systems and data flow of the new features regarding SentinelOne EDR API integration to the QaaS app project. The project is carried out as a graduation project to Christopher Sulistiyo, a 4th year ICT student in NHL Stenden as a necessary requirement to achieve his diploma.

This project serves as the first proof of concept, making it the first attempt in the iteration.

## Problem Statement

## Objectives

# Chapter 2 – Description of the Information System

## System Overview

The product in question, the QaaS app, is an ERP-like web application used by Q-ICT and its clients.

## User Stories and Uses Cases

**For customers**

As a customer, I want to be able to see the audit trail and the timeline of every threat detection and mitigation.

As a customer, I want to be able to see detections and mitigations within a specific period (week/ month) per devices.

**For the helpdesk**

For the helpdesk, the system should display more technical information compared to the customer page. They should be able to see the health status of all the devices of the customers.

**For IT user and Q-ICT**

As a cybersecurity consultant, I want to be able to offer my customers a 24/7 mitigation and checks that is included for showing real-time metrics and mitigations that are available.

# Chapter 3 Data Model

This chapter describes what data will be used in the project. It also shows how the data will flow and what are the relationships between each component in the system.

This project will utilize Firebase Firestore as its primary database. Firestore itself is a NoSQL document-oriented database. Because the nature of NoSQL databases that are designed to store data that do not have a fixed structure that is specified prior to developing the physical model, the focus is shifted on the physical data model. The developers who use NoSQL typically developing applications for massive, horizontally distributed environments. This puts emphasis more on figuring out how the scalability and performance of the system will work. But they also still need to think about the data model they will use to organize the data.

NoSQL DBs do not have a schema in the same rigid way that relational databases have a schema. There are 4 types of NoSQL database: document-based databases, key-value stores, column-oriented databases, and graph-based databases. NoSQL being the document-oriented database, typically store data in JSON, BSON, and XML format. Because the nature of SentinelOne API calls, only JSON and XML file format are the focus of the development. In a document-oriented database, documents (or items) can be nested, and elements can be indexed for faster querying.

The tables in relational databases are called collections in NoSQL database. They are the containers for documents tat share a common structure or purpose. Unlike the traditional RDBMS, collections do not enforce a schema, allowing documents withing the same collection to have different fields or structures or types. Documents are the basic unit of data storage in NoSQL, and each is a JSON-like object that contain key-value pairs. Documents are stored within collections and represent individual records or entities. They can contain nested objects and arrays, providing flexibility for storing complex data structures. Each document would then consist of key-value pairs, where the key is a field name, and the value is the corresponding data that wanted to be stored. Key-value pairs are like rows in relational database, but with more flexibility in terms of data structure. The value can be various types, including strings, numbers, Boolean values, arrays, nested objects, and even binary data.

The most widely adopted NoSQL document-databases are usually implemented with a scale-out architecture. Providing a clear path to scalability of both data volumes and traffic.

Instead of ERD, this document will present DFD to visualizes the data flow between different types of users and the data sources/ sinks.

A diagram of a company

Description automatically generated

## Schema

## User Interface Design

## System Design

## Implementation Plan

# Chapter 4 – Output

Any system is designed to produce an output, which is based on the input. In the tables below this output is detailed according to each entity and the input provided. It also goes into detail about what information users can expect to be able to get out of the system.

|  |  |
| --- | --- |
| Code Output Product |  |
| Name | Device Status and Health Information |
| User | Clients, Helpdesk, IT admins |
| Objective | To provide information and export information about endpoint () |
| Frequency | Should make an API call once every day to keep the system up to date |
| Data to be exported | XML  JSON |

# Chapter 5 Required Input

The aim of this chapter is to show the desired input. It will consist of a table for each function, that is receiving input. The table will have these categories: Code Output (a unique code for every function), Name (name of the function), Authorization (the users that will have access to use this function), Objective (the goal of the function, what data is it supposed to received), Description (where the data will be inputted on the screen), Frequency (how often will the input be put in), Screens Used (which parts of the web application (tabs) will be used when entering the input).

# Chapter 6 Menu Structure and Authorization

This project will have 3 different layers authorization: the Customers, the Helpdesk, and the Developers. Because of the security compliance regulation as the intern is not a full-time employee of the company (as stated in the Chapter 2 in Thesis), this project will only handle the GET request (Read operations) of the API call and nothing more than that, as any additional request would have a substantial adversity effect on the system.

The clients have the most basic authorization and authentication across the platform. They need to be registered on the Q-ICT database, therefore making them an official client, and register their user email, password, and phone number to the system. Once registered and the phone number verified, they can log in to the QaaS web application by entering their registered email username and password. Once they put in the right information, an OTP verification code will be sent to their registered phone number, thus ensuring MFA. When viewing the SentinelOne page, the clients can only view their own device security information given by SentineOne. The page

The helpdesk has more rights and authorization than the client user. Once logged in with 2FA, they can see the overview of all the client’s device health and status.

The IT user

# Chapter 7 Organizational Consequences

This chapter describes what problems that may occur for the users when accessing this new feature.

*How will the end-product will be tested?*

*Which conversion problems can be expected?*

*Which training courses are required for the end users?*

## Deployment Plan

## Risk Assessment

## Mitigation

# Chapter 8 Technical Consequences

*Are any extra workplaces required, and if so, with which technical facilities?*

*On which server or stand-alone computer will the software be used?*

*Is special system software or other technical equipment required to run the software?*

*How will the system be supported?*

*Which data communication facilities are required?*

*Which back-up facilities are required?*

*Who will take care of implementation and maintenance?*

# Chapter 9 Maintenance Plan

# Chapter 10 Conclusion