Penguin Technical Specification

Cievert Ltd.

26th March 2018

Version	1.0
Classification	Public

Document Control

Organisation	Cievert Ltd.
Title	Penguin Technical Specification
Author	Robert Hall
Filename	Penguin Technical Specification_Mar 2018
Owner	Robert Hall
Subject	IT Policy
Protective Marking	Public
Review date	To be reviewed annually and / or major version release

Document Approvals

This document requires the following approvals:

Sponsor Approval	Name	Date
Managing Director	Chris Kennelly	26/03/2018

Definitions

Admin -	Refers to an Application User with additional administrative rights to the system. This User has privileges above the standard Application user to redefine system behaviour
Application -	Refers to the web-based software system described in this document. This includes all core modules and any functional component described
API -	Application Programming Interface
Clinician -	Refers to any User of the Application that has locally approved access to view/review completed PROM questionnaires. This includes, but is not limited to, nurses, doctors, healthcare professionals and hospital clerical staff
Core language -	Refers to PHP, the language in which the server-side component of the Application is written
Core Application -	Refers to the central components written in PHP with any front-end (Browser) components as functional extensions, not including any of the enhanced functionality discussed
OTR -	On Treatment Review. The process of clinical staff reviewing a patient whilst receiving treatment

Patient -	Refers to any User of the Application completing a PROM assessment on behalf of a patient. This includes, but is not limited to, patients, carers, relatives, friends, parents and guardians
PROM -	Patient Recorded Outcome Measure
User -	Refers to any Clinician and/or Patient when accessing the Application

1. Introduction

1.1 General

This document is intended as a technical specification for the Application primarily from a setup and installation point-of-view. It provides an overview of hardware and software requirements, file structures, and their contents.

The initial pilot and final release Application are covered in this document as to present the system design in its entirety, along with future development and current planned limitations for the first iteration.

This document does not provide a full technical readout of each file. The Application is built and maintained as an agile platform, and therefore the software and code itself is used as the primary documentation.

Cievert believes that the information in this overview is accurate and reliable but accepts no responsibility for any consequences arising from unforeseen events. The information contained in this document is subject to change. Revisions and updates will be issued from time to time to document changes and/or additions.

1.2 Scope

The document includes an overview of the application, hardware and software requirements, including relationships between the different software and hardware components.

The document encapsulates the design at the time of writing and acknowledges that requirements, designs, minimum requirements and any other specifications as detailed in this document are subject to change.

1.3 Overview

The Application is a web-based application, written in PHP. It is accessed by the User using a standard web browser, with no additional plugins or setup required.

The Application will record the Patient PROMs independently from the clinical team. All PROM questions will be locally set by the local Admin User.

The Application will present an interface to the Patient (as detailed in section 2.2.1 Patient Interface). The interface will list several questions, to the Patient, forming the PROM.

The application will store the patient response for a given PROM event. The frequency of when PROM events will be presented to Patients will be determined locally by the clinical team. This means the same PROM question set can be presented to the Patient at determined intervals and the data captured as before. Numerous repetitions of the above process allow for the monitoring of patterns and behaviours.

From this recorded data, markers can be created that will allow any remarkable Patient response to be flagged to a Clinician for their attention.

The Application will be a web-based software application with a highly componentised architecture to support future evolution. The Application is localised as a server-based web-app, accessible via a browser for use onsite by Users.

2. Application

2.1 Application Design

The application design aims to meet the requirements as summarised in section 1.3. Core functionality comprises the minimum required to create a functional system that allows for the creation of questionnaires and the recording of patient responses. Core functionality also includes that functionality required to achieve a secure and reliable system.

The core requirements are:

- Creation, retrieval, updating and archiving of patients
- Creation, retrieval, updating and archiving of custom questionnaires
- Creation, retrieval, updating and archiving of custom questions into groups for inclusion in questionnaires
- Mechanism for grouping and sorting of questions and questionnaires into groups for assignment
- Assignment of questionnaires or related groupings to patients for completion
- Recording of patient responses to questionnaires
- Flagging of patients based on responses to questionnaires
- Authentication of users and presentation of unique role-defined interfaces

The following sections provide a more in-depth summary of aspects of the core application. In particular, each section approaches the design from a UX perspective, describing the interface and its makeup.

2.1.1 Patient Interface

The Patient will answer the set PROM questions presented to them through a process called, for the purposes of this document, patient data capture.

The Patient interface will be web-based and integrated into the main Application. This allows data capture to be supervised by a Clinician on site as required (Security models are presented in section 2.2.)

2.1.2 Clinician Interface

The Clinician User will need to create a patient profile (see also sections 2.1.3 Data Integration and 2.1.4 Data Import). The Clinician interface will provide functions for patient data capture, including both PROM and OTR. The application will feature a fully-integrated questionnaire builder. The builder will allow the creation, retrieval, updating, and archiving of PROM and OTR questionnaires.

A typical real-world questionnaire comprises several sections, each of which will include a series of questions with a limited set of optional answers for each, one or more of which must be selected by the respondent to correctly answer the question. The Application’s questionnaire builder is designed to allow the creation of a software only questionnaire in a standard format. There is also the ability to re-edit, archive, and export (see 2.2.5 Data Export) both questionnaire and subsequent answers. A separate security provision will be used to confirm authority to export any patient response data which includes a direct link to patient identifiable information (see 2.4.1 User Authentication).

2.1.3 Data Integration

Data integration and data import operations will be made available through the use of an extensible API interface.

APIs will allow data to be imported from systems not specifically designed to operate with the application. The API will accept a single patient record or complete upload file containing multiple patients in a specified format. Custom APIs can be added based on future requirements.

A PDF exporter will allow complete episodes of data to be exported ready for print as a standalone document or for upload onto third party systems such as electronic patient records. In addition, a separate customisable reporting module to group like-responses together for analysis will be available. Export file formats include CSV, PDF and/or an Excel compatible spreadsheet format.

2.2 Application Security

2.2.1 User Authentication

The core application's security is an extension of Laravel 5.5 security functions. An ACL (Access Control List) will be built on Laravel's foundations in order to grant access on a per-right basis. User rights are collected into manageable groups for assignment to users with similar privileges.

Authentication in each case will be required to validate either Patient or Clinician User. This pre-assigned user access level will restrict access to sensitive parts of the system. A Clinician will need to enter a valid user email and a valid password linked to that email address in order to gain access to the system. A Patient will need to have a valid episode within the Application, created by a Clinician and will then need to independently validate their identity to access their predetermined PROMs. To do this they will be required to enter their date of birth and gender as a minimum requirement.

2.2.2 Database security

The database will not be available directly via the Application's interface. Access to the database will require server access at a minimum, discussed in section 3.5 Server Security. Any data presented will be the result of pre-defined queries that follow all standard guidelines for safety with regard to SQL injection. This is achieved by using Laravel's dedicated libraries for data access within PHP that are built upon PHP's native PDO modules.

2.2.3 Data Export Security

A separate user right will be configured to control authority for the exporting of reports or other exportable documents that link patient questionnaire responses to patient identifiable data such as names, unique identifiers (NHS, Trust Numbers, etc.).

Ultimate authority for assigning the above right resides with any user granted administrative system rights.

3. Architecture

3.1 System Architecture

The core Application will require a suitable web server architecture, including HTTP server, MySQL database server and Application server for dynamic content.

3.2 Application Architecture

The core Application is written as a web-based application. The primary server side language is PHP complimented with a MySQL database for information storage and retrieval. The client side code is written in JavaScript, CSS3 with complimentary open-source libraries. The Application was designed with the goal of achieving maximum usability from the outset. Provision is made for the use of all modern browsers and only stable third-party libraries are used in client-side code.

The architecture will be built on the open source Laravel PHP framework which provides a base MVC infrastructure for the development of server-side web-based applications, including template-engine, authentication and session management. The primary data store for application data will be MySQL, an RDB.

A single virtual server will be used to house all server components, including Web Server, Database Server, and Application Server. Libraries will be required and come installed as part of the build. These include the following:

• Laravel	v5.5 (or above)
• Bootstrap	v4.0 (or above)
• jQuery	v3.3 (or above)
• Require.js	v2.3 (or above)
• InteractJS	v1.3 (or above)

3.3 Minimum Requirements

The following specifications are required to run the Application:

Software:

• PHP	v7.0 (or above)
• MySQL	v5.7 (or above)
• Ubuntu	v17.10 (or above)
• Nginx	v1.13 (or above)

Hardware:

• CPU	2GHz or above, 2x virtual CPUs (dual-core)

• RAM	8GB
• Disk Space	The application requires around 20-40GB to store information, but a more sensible requirement of 150-200GB ensures smooth archive and backup capabilities

3.4 End-user Requirements

The Application has been built using HTML as a mark-up language to dictate structure and semantics, CSS for styling and visuals, and Javascript for effects and additional functionality. It complies with all current standards as dictated by the W3C. The application runs best on any PC or Mac using a web browser with a screen resolution of at least 1024x768.

In theory, all web browsers are supported; but for compatibility and security reasons the following browsers are recommended:

- Internet Explorer – all versions supported by Microsoft
- EDGE – all versions supported by Microsoft
- Firefox – all versions within 2 years of latest release
- Safari – all versions within 2 years of latest release
- Google Chrome – all versions within 2 years of latest release

The Application is also supported on iPads and other tablets, but is not currently suitable for use on mobile phone.

3.5 Server Security

Access to the server will be granted to Cievert Ltd. development staff only via SSH secure authentication. Deployment of code will be made manually and under full control of the development staff. This means no third-party deployment tools will be used to update live instances of the code. In addition, all databases used for the application will require authentication with username and password.

Cievert uses password supported SSH encryption rather than a plain key since access to the server may give access to the database where the application's source code and database are housed on the same server.

As an additional note, the PHP code that drives the system is housed outside the web-root directory so incorrect configurations will not risk exposing the code content of the web application.

4. Backups

4.1 On-machine Backups

The Application is backed up automatically onto the local machine into the /srv/backup folder. Daily backups for 7 days, weekly backups for 4 weeks, and then ongoing monthly backups are kept of both the application and the databases.

These are stored on machine for quick resolution in case of any issues with the Application data set.

4.2 Off-machine Backups

Off-site backups can also be configured, and can be either managed by Cievert or by the VM provider as required.

4.3 Faults and Response Time

Made with reference to section 5 within each individual Penguin software contract, the disaster recovery plan with Casper is as follows:

Faults will be categorised as urgent or standard by the Supplier in accordance with the following definition. Where a fault adversely affects the integrity of treatment referral data and/or patient demographic data stored within the Software (**Urgent Fault**), the Supplier shall use reasonable endeavours to provide a fix or work around within one whole working day of receipt of notification of such fault. In respect of all other faults (**Standard Fault**), the Supplier shall use reasonable endeavours to provide a fix or work around within 40 working days.

In the event of server failure, the VM provider must make reasonable efforts to provide a new or fixed server instance as soon as possible, and this timing will not affect the response time of the Cievert team within the urgent fault classification described above.

4.4 Code

Code is stored in a remote repository on secure servers within the EU in line with current legislation and best practice. Different versions are available at any one time and can be recopied to a client server in the event of a server failure resulting in loss of data. This is code repository only and no patient identifiable data is stored with the code.

4.5 Data

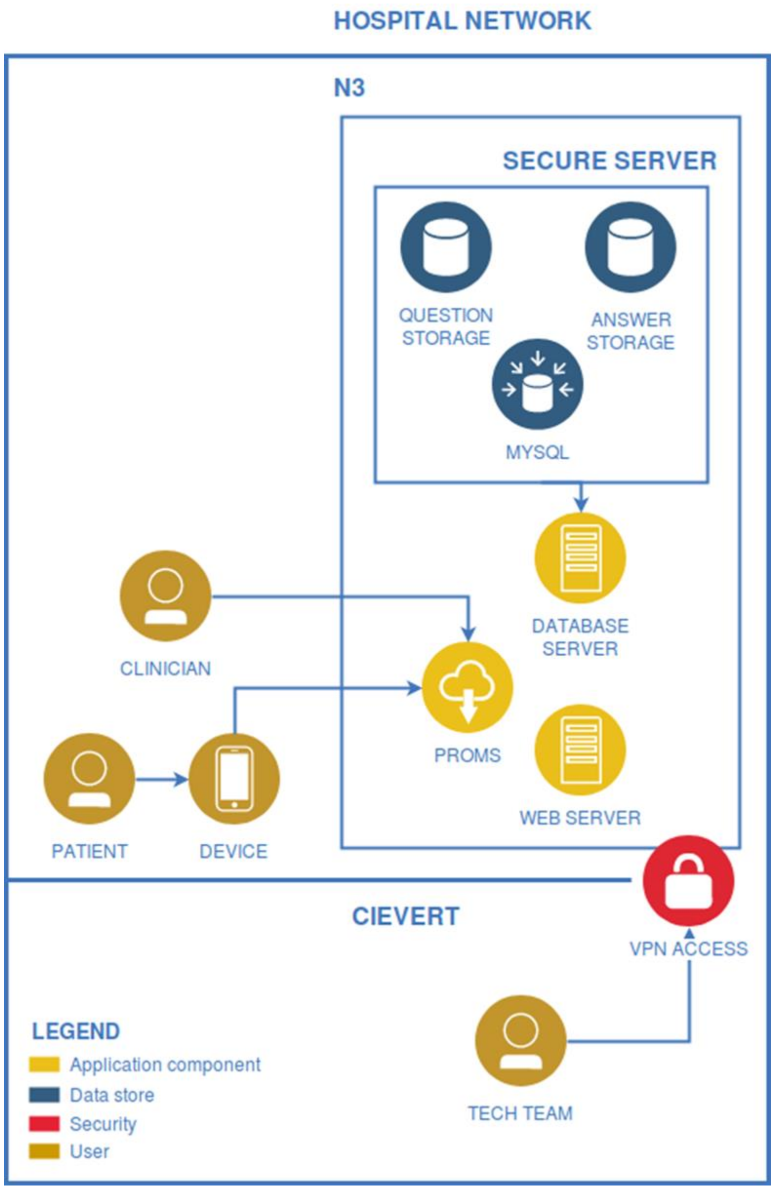
Databases are updated continuously. In the event of errors occurring due to server problems or the otherwise incorrect or incomplete processing and storing of data, local server backups can be used to restore the system to a previous given state.

In addition, the server used for client instances has a dedicated backup solution that will store the entire contents of the server, including databases.

5. Network Diagrams

5.1 Local Access N3 Model

The following network diagram illustrates the Application hardware and network setup within both the Cievert hosted solution.


HOSPITAL NETWORK
N3
SECURE SERVER
QUESTION STORAGE
ANSWER STORAGE
MYSQL
DATABASE SERVER
CLINICIAN
PROMS
WEB SERVER
PATIENT
DEVICE
CIEVERT
VPN ACCESS
TECH TEAM
LEGEND
Application component
Data store
Security
User