Introduction

At the moment the GemsTracker Development Team is searching for suitable hosting companies that are interested in gaining expertise to quickly implement new GemsTracker (GT) installations and that are able to expand the capacity of existing installations on demand. We wrote this document to help hosting companies in the setup of GT installations.

By preparing a couple of standard virtual server images for GT installations we hope to bring down the installation and maintenance costs of each individual GT installation.

Gemstracker

GemsTracker, the Generic Medical Survey Tracker, is an open source web application available from www.gemstracker.org and GitHub. On the GT site you will find more information and examples about how GT can be used in research and clinical care. Moreover, you find more user en developer documentation on our wiki (http://gemstracker.org/wiki/doku.php?id=start), which also contains a quick start for setting up GT sites. GT consists of a core library with a default installation, but is designed from the ground up to enable developers to extend the core functionality for specific projects. There are currently more than 30 different projects in production, each comprising a different website. The projects range from having only the most minimal project code of maybe 20 lines of code; up to the Pulse and ZSD projects that almost rival the GT library in their project specific code-count.

This document describes the requirements for webservers hosting one or more GT installations. It addresses security and explains the GT architecture in order to explain the different deployment scenarios. The document then gradually moves on to more detailed configuration information.

Security

GT is an application for collecting survey answers containing personal patient data using the web. This is privacy sensitive information and therefore security is an important consideration. In most countries one is required by law to make a serious effort using the best industry practices to keep the information inaccessible to people without a contractual obligation to respect the privacy of the patients.

In the Netherlands the best practices are documented in the NEN 7510 standard. NEN 7510 certification requires that the hosting company should at the very least be ISO 127.0001 certified, therefore a hosting account on a shared webserver is usually insufficient for certification purposes.

The decision to NEN 7510 certify a particular project is up to the owners of that individual project, but in our opinion the default hosting and application environment should pose no obstacles to certification.

Architecture

GT is a library that enables health care staff to make sure the right questions about the right patients are asked at the right time. What GT does not do is asking those questions. For survey creation and entry GT uses third party products through a pluggable architecture. This gives hosting companies the freedom to also use their own (proprietary) questionnaire systems and benefit from the GT software functionality. Currently GT has plugins for the LimeSurvey and OpenRosa and at least one project has a project specific survey plugin. LimeSurvey is the most used survey plugin, but where LimeSurvey is mentioned in the rest of this document the intention is that LimeSurvey can be replaced by another survey sytem.

Patient identifying information is stored by GT, together with an anonymous random patient identifier that is hidden from the users in the GT interface. This identifier is also stored in the questionnaire software (e.g.LimeSurvey and OpenRosa) with the answers belonging to the patient. Preferably all treatment data is stored in these survey sources in databases separate from the database used by GT. This has the advantage that during the treatment and/or the research GT can show the treatment data for each patient, but during group analysis and evaluation the data cannot be traced to individual patients.

GT installations can also create a bridge table for data warehousing containing all the answers from all the sources in a format easy to use by data mining packages. Again this table can and should be stored in a separate database in order to keep treatment data and identifying information separate.

Source code

The GemsTracker open source code is stored on GitHub. Individual projects should also use version control for their code storage with links to tagged versions of the open source code. In a preferred deployment all source updates of the production system should be from tagged versions.

Application environments

Testing or developing a GT application usually requires different project settings than during production. Together these different settings form the application environment of an installation. In the Zend Framework this environment is primarily determined using an environment variable in the underlying operating system. The application environment determines usage restrictions, caching, folder locations and utility application. E.g. all GT environments except the production environment bounce the e-mails generated by the GT to the sender instead of the test recipient. The reason for this is that all these environments are used to try out the software. When sending an e-mail the user does not want to actually send the e-mail to that address, but instead wants to test how the e-mail looks for the recipient.

These differences in behavior are why GT always tells a user in which environment her or she is working. To set these modes the Zend Framework uses the operating system APPLICATION_ENV environment variable. GT projects can use these application environments:

APPLICATION_ENV Description production Default. For production development, highly cached and high security. Demo A demonstration/learning installation, lower security and e-mail bouncing. acceptance Identical to production except for e-mail bouncing, but shows that it is a different environment. Testing General testing, debug output, lower user password security for easy testing and e-mail bouncing. development Lowest security and e-mail bouncing, by default without any caching.

Multiple versions of the same project with different application environments can be installed on a single server using server specific settings e.g. in the .htaccess file or by using url parsing code in the index.php file of the application.

NEN 7510 certification requires the installations containing the testing and development environments to be on different hardware than the production environment. On the other hand: there are no objections against a demo environment on the same server as the production environment. The acceptance environment can be on the same server as long as a new version is first tested on separate hardware. Of course in all cases the content of the data in the production environment must be separate from the data in any other environments, both when the data is on the same hardware and when it is on a different system.

Deployment Scenarios

A GT deployment always consists of these parts: - The code specific for the project. - The GT code. - The Zend Framework code. - At least one survey source installation. - A MySQL database server and a (possibly different) database server for the survey source. For all current source types the GT installation needs to be able to directly access the database of the survey source – this does not have to be a MySQL database, though a database engine supported by the Zend Framework is necessary.

The survey source itself can be installed on a different server, but this is not required. Usually a subdirectory of the GT installation is used or the source is installed using a separate sub domain. Here are some example url's. When installing GT on a server with an existing site, it may look like this: - www.project.url/gems – GT installation - www.project.url/ls – LimeSurvey installation When the project server only uses GT and LimeSurvey this is the usual installation: - www.project.url – GT installation - www.project.url/ls – LimeSurvey installation LimeSurvey is installed in a subdirectory as GT is the default url where users should go to. Neater is an installation using sub-domains for the different installations: - gems.project.url – GT installation - survey.project.url – LimeSurvey installation

Using sub-domains may be costly in terms of SSL certificates, as all url's should be used over HTTPS/ Working with sub directories using a common root allows the use of a single standard SSL certificate. However, due to the design of GT individual sub-directories should not be real subdirectories of the web-root. Instead symbolic links or virtual directories should be used – as will be explained later. We hope this demonstrates that GT deployment is flexible and can be adapted to different needs and to the infrastructure available.

Single project – small or medium

For projects tracking up to a thousand patients a simple setup is usually sufficient. Of course when those patients have to answer ten surveys each day it might be insufficient, while conversely it might be sufficient for a project with ten thousand patients who have to answer a single survey each year.

In these cases a single (virtual) web + database server is probably sufficient, though using a separate database server might be preferable for security reasons.

Still this setup is usually complicated by the fact that the same server often has to contain a second and possibly even third GT installation of the same project but using a different application environment, e.g. the demo or acceptance application environments.

Another possible complication is the installation of other web applications on the same server, e.g. a CMS. This is not an issue on large projects or multi projects installations where the requirements demand that other functionality is installed on separate servers, but for small projects it is not uncommon that a small website is required to explain the project and/or the organization behind the project. Of course another application should only be installed on the same server when it does not block NEN 7510 certification where it is needed. I.e. the GT database data and setup directories should not be accessible by this application.

To summarize: even small project servers should be set up in such a way that multiple applications can easily be setup. E.g.: - www.project.url – cms installation. - www.project.url/demo – demo environment - www.project.url/demols – demo LimeSurvey environment - www.project.url/gems – production environment - www.project.url/ls – production LimeSurvey environment - www.project.url/phpMyAdmin – IP range limited and password protected phpMyAdmin or another database administration console

Single project – large

For large projects with a heavy server load the first step is to separate the (virtual) database server from the (virtual) web server as the technical requirements of web and database servers are different in that a database server primarily needs a large memory and quick storage, while the primary webserver requirement is processor speed.

When further capacity is needed, there are two optimization directions that do not require any code adjustments. The first direction is to split the survey system web server from the GT server. A single survey system installation can – at least in the case of LimeSurvey - be further divided into multiple installations on separate servers. The central information connecting the surveys and answers in different survey installations to a particular patient is after all stored by GT.

The second direction is to use multiple database servers. A standard GT installation uses at least two databases: one for the GT data and a second for the data in the survey source. When using multiple survey sources each source can use a separate database server. Another separate database server may be employed when an extra database is used to provide data to a data mining application.

Any further optimizations would need adjustments to the core GT code. So far this has not been necessary, but adjusting GT e.g. to employment in a webserver farm should be possible.

Multi project

While some GT projects are huge, most are limited in scope. What those projects need is a simple solution to be hosted, e.g. on https://www.hoster.org/project1, with an option to be able to use a

project specific full url: https://project.organization.org/.

In this scenario multiple GT projects are installed on the same server. A default multi project server configuration would consist of three virtual servers: a production server, a database server outside the DMZ for the production server and a demonstration / acceptance server running a copy of the code on https://demo.hoster.org/project1, where all code should be released first on the demo server and only afterwards should the code be installed on the production.

The demo server can also be it's own database server as the workload of this server is usually minimal. A default new project should be installed with working versions of both GT and LimeSurvey (or another survey system) including separate databases for them. It would be nice to have a simple CMS solution available as an option and PhpMyAdmin or comparable access to the project databases should also be an option. The server should be setup so that new GT versions and patches to existing GT versions can be centrally installed and updated. The same goes for new versions of the Zend Framework and LimeSurvey. We are currently looking into Zend Server as a possible solution for some of these deployment issues and to add an extra security layer for auditing changes to the server.

A server of this type would require some extensions to the GT code, like a setup script for creating new projects and shared GT libraty files. However, the main problem to be solved would be the billing of the users as we see this as a server where people can sign up and get started without extensive consultation, but that doesn't mean everybody can sign on for free.

Database security

While some projects install all the project data in a single database, this is not the preferred storage configuration for GT. A safe GT installation consists of two or three databases. The number of database users will be the same, but their configuration will be different as their will not be one user for each database, but rather the different users have different roles in accessing the database.

There should be a project_gems database containing the patient identifying information, at least one project_ls database for each LimeSurvey instance used in the project. When the data should be exported to a data warehouse the extra project_data database is needed.

The database user for the GT application must have read/write access to all the databases. The user(s) for the LimeSurvey installation(s) may have only read/write access to their LimeSurvey database. Actually multiple LimeSurvey installations can share a single database using different prefixes for their tables, but we advise to keep this data in different databases. The optional data warehouse user should have read-only access to both the project_gems and the project_gems_data databases.

Database Accessible by user Description project_gems Gems, Data warehouse (read-only) Contains identifying data project_gems_data Gems, Data warehouse (read-only) Optional, contains medical data for data mining project_ls Gems, LS Contains medical data in survey answers

When the installed CMS uses a database, then that database should be isolated from the GT databases and the CMS database user should have no access to GT data. The reverse is not required but is good practice.

GT, LimeSurvey and Openrosa in conjunction with GT can all build their own database structure using their web-interface, but the databases should exist prior to initialization.

The best practice for the GT database is to define the database server, name, user and password in the file var/settings/db.inc.php.

Email transport

GT applications can send out heavy mail loads and servers may not hinder these mails and should do what they can to make sure the mails are not seen as spam.

GT has built in functionality that routes email through different servers depending on who is sending the e-mail – including secure login when required. E.g. the Erasmus MC does not allow any @erasmusmc.nl mail to be distributed from other servers than their own, so external GT installations sending mail from Erasmus MC users have to login to their server to get the mail distributed. GT does this, though unfortunately it requires storing a password in the database.

In case of organizations that do not allow remote login for sending mail and that do limit the locations their mail is sent from, the IP address of the location where the mail is sent from should be added to the list of allowed originating mail IP addresses. In all other cases the provider should monitor mail blacklists to make certain that the GT server does not appear on them and take action when this does happen.

Hosting GemsTracker

GT 1.8 is written in PHP 5.6, uses a MySQL 5.1 or higher database and works on most webserver platforms including Windows/IIS, Windows/Apache, Unix/Apache and Unix/Nginx. The application is built using the Zend Framework version 1.

Requirements and tasks for hosting companies: Hosting companies interested in providing GT services should be able to meet at least these minimum requirements: - Experience in hosting for healthcare or else at least in scientific research. - Knowledge of privacy and healthcare laws and regulations. - An experienced technical helpdesk, reachable by phone. - A flexible range of products from ‘Do-it-yourself’ server installation to managed server hosting. - A flexible range of service levels from 8 x 5 to 24 x 7 and from 98% to 99.99%. - ISO 127.0001 certification for storage security. - Storage management, e.g. advance warning when disk space threatens to run out. - Flexible database / server backup solutions. - A high level firewall with at least IP address level access restrictions. - Email management, i.e. blacklist tracking and warning for sites hosted by the company. - Certificate management for the applications. - PHP and MySQL hosting experience.

Preferred requirements are: - Being able to manage application deployment e.g. through Zend Server or SVN. - Zend Server or comparable deployment experience. - NEN 7510 certification experience. - Email management including activities to prevent blacklisting. - PHP development experience.

Lastly we would prefer companies able to provide consultancies services to parties starting with a GT project, either directly or through third parties. A separate document for consultancies is in the make.

Technical requirements for hosting GemsTracker

There are no hardware requirements; or rather there is a strong preference to using virtual servers so that we can quickly adjust the hardware to the requirements of a specific project. So we specify only the software requirements for hosting GT, both minimal and preferred.

```
Minimal (Strongly) Preferred
```

OS Web Server Windows 2008, Unix Windows 2012, Unix Web server One of these: • IIS 7 with ReWrite module • Apache 2 with mod_php and mod_rewrite • Nginx One of these: • IIS 8 with ReWrite module • Nginx OS Database Windows 2008, Unix Windows 2012, Unix Database server MySQL 5.1 Community MySQL 5.6 PHP 5.3.x 5.4.x PHP Modules Default PHP Modules plus: • Curl • Dom • Fileinfo • JSon • GD2 • Multi-Byte String • MySQLi • MySQLnd • SOAP • XMLReader • XMLWriter Minimal PHP Modules plus one of these: • APC • Memcached • XCache Zend Framework Version 1.11 or higher, but not 2.0 or higher The newest stable 1.x version Zend Server n/a Zend Server Professional URLs One Multiple SSL Always required for each URL Always required for each URL Mail server Required Required SVN Preferred Preffered

Directory structure

The GT code is by default installed in multiple sub directories, only one of which is accessible from the web. The other code, library and storage directories should be stored in a location accessible by the web server, but not from the web. - application – The project specific code. - htdocs – The web root directory containing index.php, JavaScript and stylesheets, and possibly code for survey sources. - library – The GT code. - var – writeable by the web server; contains cache, uploaded files and server specific settings e.g. for database access. Other directories you may find on the server: - scripts – Scripts for the application, e.g. for command-line invocation of GT. - test – Unit tests, not needed on deployments, but part of the development environment.

Unless specifically told otherwise in the index.php file, GT applications assume all these directories to be in the same parent directory, including the htdocs directory – though this is the one directory that can have a different name without breaking the code.

As this structure would cause problems with multiple installations running from subdirectories on a webserver, the usual deployment scenario is to link the htdocs directory to the web root of the webserver using symbolic links with Apache (on Unix using ln and on Windows using mklink) and virtual directories with Windows IIS. PHP.INI settings

In general the standard php.ini settings for production and development servers suffice for the production / all other environments. However, some non-standard php.ini settings should be set. Setting Example Comment Per project alternative date.timezone Europe/Amsterdam Should be set to the time zone the project should use. In the index.php of the project. error_log /tmp/php-error.log Must already be set on php startup. Override in the project’s application.ini. Starts with the php.ini log, switches during startup to the project log. include_path /path/to/zend/1.x Can be placed in the projects sub-directory, but sharing is preferred. In the index.php of the project. soap. wsdl_cache_dir /tmp/ Must be set to writable directory (when used). Override in the project’s application.ini. upload_max_ filesize 60Mb The maximum size for uploaded survey PDF’s Override in the project’s application.ini.

Scheduled jobs

GT projects use cron or Windows Schedule for automated tasks. Currently these jobs usually run twice a day at 7:00 o'clock and 19:00 o'clock, but we are developing a sub-scheduler within GT to enable different jobs to run at different moments in time/ These jobs will still be started from the central GT cron function that should then run every 15 minutes or whatever time is appropriate for the specific project.

GT can be started directly from the command-line so no in-between application like wget is needed. All the scheduler needs to start web root index.php with PHP and with the parameter cron: php –f index.php – cron Of course the location of the PHP executable and the index script must be added to this command.