

HCI590 – Independent Study in Database Management Systems

Project Design Decision Document

This document outlines the preliminary design decisions that have been made to address all features that can be found in the requirements document. In the following, every critical software feature is listed. The accompanying paragraph shows the design choices that have been made to accommodate this feature in question.

Automatic Content Tracking.

This feature is intended to provide a clean and easy way to track the contents of a website. A system is desirable that is as simple to use as possible and works with as little interference as possible. To address this problem, the idea for a plug-in for commonly used web server software products has been evaluated, but was soon declined. Reasons for this decision were that it is too complex to provide plug-ins for the wide variety of web server solutions and that limiting to a few, most common products is unacceptable for this project. More importantly, a plug-in would require administrative access to the web server software, which is also not desirable for this project.

To address this feature nonetheless, a custom light-weight Java software is going to be implemented that can parse a directory structure and automatically keep track of all necessary components and data. This solution has the advantage that it can be scheduled to run independently from the web server process and no administrative rights are necessary on the host computer to execute this program. This makes this project software usable by people that only have guest or simple user accounts on machines and want to keep track of their personal websites.

Remote Data Collection.

In order to track visitor information of a website, it is necessary to collect data from the users of the website. This is easiest done by simply extracting information from the user's browser using simple JavaScript or PHP technology and adding it to the database on the server. However, this poses two critical problems. Firstly, it would require the user of this project software to have massive knowledge in web programming, as it is necessary to make significant changes to the html documents the user wishes to incorporate into the tracking. Secondly, this poses a security issue, as database manipulation would occur on client-side rather than on server side. To work around both issues, a simple client-server architecture will be implemented. A collection of JavaScript functions that can easily be incorporated by adding one single line with a link to the source file to every html document that shall be tracked will be implemented. These functions will collect some data about the user of the website and send it to a light-weight custom web server that only receives html POST data and adds it to the database. This way, database access is merely on client side and adjusting the html documents can be held to a minimum.

Automatic Representation of Data.

There are essentially two ways to represent the data from the database so that it can be inspected by the user. A Java-based software can be written that periodically parses data from the database and outputs a html document with the represented data. This has the advantage that it reduces the server load, as access to the database is only necessary, when the Java client is running. This can be set to a low threshold. Disadvantage of this approach is a lack of versatility. The user would be confined to the output that the Java-software is meant to return. More dynamic is the other approach. A collection of

PHP -enhanced html documents will parse the database on demand and return the desired information. This way, the user has control over what is being displayed, but the server load is heavily increased. As both approaches have considerable advantages and disadvantages, it has been decided that both approaches will preliminarily implemented. Support for one of the two approaches might be dropped further on in the project.