Account This!

Group 9

Kristoffer Renholm
Johannes Edelstam
Joakim Ekberg
Jesper Skoglund

Project Overview Document

1. Who are the users and what problem does the system solve for them? ¶

Account This! aims to deliver a bookkeeping system suitable for small companies. Small companies often has a small budget where every cent matter. Furthermore small companies does not have a lot of knowledge in bookkeeping, therefor they need a bookkeeping system which is easy to use. We have discovered that there is very few products which can offer both. Small companies wants to spend their time with their key products, not on bookkeeping. Such companies could for example be hairdressers (almost every hairdresser in Sweden has a one-man-company which rents a hairdressers chair in a barbershop), consultants or similar small companies. Many people has a company in addition to their full time work, they in particular does not want to spend a lot of money and money on bookkeeping.

2. The main uses of the system. ¶

Account This! will be a system for double-entry booking. This is the standard in most businesses and organizations. But Account This! will be a web-based system with an easy to use user interface. Therefor the main use will be an easy way to do the bookkeeping in a small company. Account This! will not suit larger companies because we want to keep it simple. Otherwise small companies will not use it.

Use case 1 ¶

Peter runs a small painting firm. He has just received payment for a job. He logs into Account This! and clicks on “New voucher”. Now he sees a table view. He adds a row which debts one account of his choice and adds another row which credits another account. He clicks create and he is done.

Use case 2 ¶

Peter suddenly realizes that he made a mistake, the sum of the invoice he added was incorrect. He finds his previous voucher in the voucher list and clicks edit. Due to Swedish law you cannot edit a previous voucher, instead the system creates a new voucher which overrides the old one. Peter can now make the changes needed.

3. The context/environment in which the system is to be used. ¶

From the user points of view the system will only be accessed through the user’s web-browser on an internet connected computer that support the required web-browser. In this context the only requirements of the environment should be that the user uses a modern web-browser. In our case a modern web browser should be one of the following three:

- Firefox 2.0
- Internet Explorer 7

- Safari 3

This is required because we want to keep the testing process as small as possible, however, if a user wants to use an alternative browser, suppose opera, we allow this but we cannot guarantee that everything works. This environment is available almost everywhere where there is an internet-computer available, for example at work, at the library or at home.

4. The scope of the system. ¶

To make the "Account This" project successful we need to meet more goals than what's possible to do in this course. This is due to time restrictions and manpower. We would need much more time of development. Because of this we just select the most important subset of features to get a great prototype.

We want do develop the most basic accounting application there is and put it online. This platform should then be possible to extend in a simple matter.

This prototype, as I said should support the most basic features of an accounting application. These features are presented below in an in-out chart.

Topic	In	Out
Add voucher	X	
Handle the way a verifikat is changed according to all laws that apply	X	
Manage different accounts	X	
Able to import the 'bas' - chars of accounts.	X	
Import data from other systems according to the standard SIE-format. This format could easy be exported from, for example, the well known SBCS Accounting Program.	X	
Generate different types of reports for example Balance-Report.		X
Basic authentication using username and password	X	
Extensive authentication using other methods verifying the user's identity, i.e. using one time codes.		X
Extensive logging of what the user does within the system.	X	
Every operation in the system should use transactions so no data are lost	X	

5. The main factors that need to be taken in to account when designing and building the system.¶

Whilst an online-based bookkeeping system must be rigid to secure that the inputs act in accordance with the rules and laws of bookkeeping (Riksdagen 2007), it must also be customizable to ensure that enterprises can adapt the nominal ledger and the general journal as it desires. This implies that the system should be built in a way that allows user specific adjustments and expandability, taking the following factors into account:

- Inaccurate transaction data and/or erroneous numbers could have fatal implications to the user. This would not only generate user frustration, it could also result in costly lawsuits that would hurt the reputation of the system.
- A fundamental idea behind the design of the system is its high degree of accessibility. The possibility for a user to reach his/her bookkeeping from anywhere requires: (1) that users are able to login to the system using a typical web browser along with his/her username and password; (2) that the system is able to handle electronic traffic from computers at various geographic locations; and (3) that no unorthodox support applications need to be download to use the system.
- Storing transaction information in the system would do no good unless it could be made explicit in several ways. The system must be able to export information to printers and other applications such as Excel. An optional solution could be to allow the creation of sub-accounts with limited (observer) functionality.
- The information a user enters into the system contains financial enterprise data and should thus be regarded as confidential. It will consequently be necessary to store and cipher sensitive data and handle it with a high grade of security.
- Another important issue facing an online bookkeeping system is the two topics of speed and reliability. Users are unlikely to use the bookkeeping system unless it's as quick and reliable as other bookkeeping software. The issue of speed implicates that the system is built to support short loading times, that it's designed without unnecessary steps and that the bookkeeping is easy to navigate. The issue of reliability means that users should be given no reason to doubt the uptime of the bookkeeping system. Neither connectivity problems nor application breakdowns can be accepted.
- As different users may desire different bookkeeping methods, the online bookkeeping system should support several account plans – ex. BAS (http://www.bas.se/) – used in double entry bookkeeping.
- It must be possible to sort verifications and alter whether data is shown alphabetically or chronologically.
- To avoid the handling of frustrating mistakes, users should have the ability to commit or reject the changes they've made. A useful but not necessary addition to the system

would also be a history tracking tool that allowed users to make leaps in time by undoing (or redoing) changes.

6. Technologies and Risks ¶

Developing web-based applications can be very time consuming as of the large number of technologies involved. For example, as a professional web-based developer you are required to master at least:

- HTML – Hyper Text Markup Language, a technique for displaying user interface over the Internet.
- CSS – Cascading Style Sheets, language that is commonly used to describe the presentation of a document written in HTML.
- JavaScript – Scripting language used to enhance user experience by allowing code to be executed on the client side of the application, the browser.
- A programming language – To allow users to interact with a webpage a programming language is used on the server side to handle logic how a page should be displayed and maybe connect to a database to fetch information.
- SQL – Structured Query Language, user for creating, updating, deleting and selecting data from a database.

By using a web-application framework, some of the technologies mentioned above can be handled by the framework. This simplifies the development of new web-based application when for example SQL are handled and generated by the framework. Most web-application frameworks will also provide security features and other convenient helper functions.

In this project we are planning on using the Ruby on Rails web-application framework. Ruby on Rails will provide us with many features like:

- Object-relational mapping.
- Application skeleton – a pre decided folder and file structure.
- A clean Model-View-Controller architecture.
- Easy to use programming language Ruby.
- Possibility to use pre-packaged functionality. For example a plug-in called acts_as_authenticated may be used for authentication of users.

Whilst there are many positive features using Ruby on Rails, there exist some risks:

- No distributed transactions – needed for making transactions over multiple databases. This will not be a problem at first but may cause trouble in the future if the system is required to modify an external source.
- Sometimes hard to find references and help with problems. Much of the Rails-knowledge on the Internet is found in badly indexed blogs. This will have an impact on productivity.

- Slow – for a large user base Ruby on Rails will not be very efficient because of the relatively small number of requests per second handled by a mongrel web server. This can be solved using multiple mongrel instances but with an overhead for each server.

7. References ¶

Riksdagen (2007). Svensk författningssamling (SFS). Accessed on 11/11/2007, http://www.riksdagen.se/webbnav/index.aspx?nid=3911&dok_id=SFS1999:1078&rm=1999&bet=1999:1078