FinalReport

Bergar Simonsen

December 16, 2012

Contents

1 Business Modeling

1.1 Vision

1.1.1 Introduction

Our goal is to make an interactive document sharing system, Slice of Pie, which allows multiple users to easily share and edit documents both online and

1.1.2 Problem statement

Sharing and editing documents can be cumbersome. Sending a document back and forth between multiple users can lead to a lot of errors. Users can overwrite what another user has done, and if they aren't all using the same text editing system this can lead to formatting issues in the document.

1.1.3 Summary of system features

- Multiple users must be able to share and edit documents online.
- Synchronization for offline usage.
- Merging of documents.
- History. Which allows the user to see all recent changes made to the document.
- Documents can be categorized into folders or projects in order to get a better overview when working on a larger project with multiple files.

1.2 Glossary

- **Response Time:** The time it takes for the system to respond to a request from the user.
- **Document:** A document refers to a complete document, not just a single file. A document contains a owner, id, content and a file.
- **Client:** A client can mean two things. A web client, operating directly on the server, and a stand-alone client, that runs offline, locally on the end users machine synchronizing with the server.
- **System:** Refers to the core of the application. This includes document handler, user handler etc ..

2 Requirements / Analysis

2.1 Use cases

- UC1: Create new document
- UC2: Edit document

- UC3: Delete document
- UC4: Merging documents (resolve conflict)
- UC5: Offline sync
- UC6: New folder
- UC7: New project
- UC8: Find old version of document
- UC9: Share Document
- UC10: Log in

2.1.1 Use case Diagram

2.2 Supplementary Specification (FURPS+)

- Functionality
  - The system must be able to create/edit/delete users.
  - The system must be able to create/edit/delete/share documents.
  - The system must keep a log of all document actions.
  - All system usage requires user authentication.
  - The system must support multiple users.
- Usability
  - The system must be easy to use.
    - Have a clean user interface.
    - 8 out of 10 users must be able to use the system without any training.
  - The system must be easily visible for people with "not perfect" vision. E.g no graphics that blurs the view of the core system functionality.
  - The web client must be easy and quick to navigate. No function should be more than 3 clicks (windows/sub-windows) away.
    - Not counting navigating a users files.
- Reliability

  - It must be possible to use the system without any internet connection.
    * With some limitations.

- Performance
  - The system must respond instantly
    * A request must not take more than a few (2-3) seconds.
    * Not taking external factors (such as bad internet connection) into account.

2.3 Domain Model

INSERT DIAGRAM HERE

2.4 Logical Architecture

INSERT DIAGRAM HERE

2.5 System Sequence Diagram

INSERT DIAGRAM HERE

2.6 Operation Contracts

INSERT DIAGRAM HERE

3 Design

3.1 Class Diagram

INSERT DIAGRAM HERE

3.2 Interaction Diagrams

INSERT DIAGRAM HERE

3.3 Technical Memos

3.3.1 File Format

Issue: Files format - Which file format to use **Solution:** Summary: Use HTML for our file format. **Factors:**

- Must be able to contain both text and

Solution: We chose to use HTML for our file format because it's simple to construct, and can contain text and images seamlessly. **Motivation:** We needed a file format that can contain images and text as well as being easy to construct. in addition, HTML can easily be extended to other content. Lastly, HTML can be opened with any browser, so the users isn't tied to SliceOfPie if he just want's to view the content of a file. **Alternatives considered:** We considered using a .txt file format, but .txt can only contain plain text. We also considered using our own file format (since the format itself isn't important to the application). But if we use our own format the user is stuck with using SliceOfPie, so he can't view the content of a file with any other application.

3.3.2 Document Merge

Issue: Merging two versions of the same document. **Solution Summary:** Git-hub inspired merge. **Factors:**

- Merging two versions of the same document without overwriting existing changes.

Solution: Our merging algorithm reads the two documents and stores them, line by line in an array. Then the algorithm compares each line in the two arrays, if the lines are the same, insert the line into a new array. If the two lines aren't identical, insert the new line into the new array + insert the line from the old array in the next line. This line will be encapsulated with <<< TEXT >>> which shows the user where there is a conflict which the user can solve later on. If the new version of the document contains lines that aren't in the old array, they are simply added to the new array. **Motivation:** There are other, more advanced, merging algorithms available. Because of time constraint we chose to use this one. It isn't the most advanced/complete algorithm but it does the job quite well considered it's simplicity. **Unresolved issues:**

- Our algorithm doesn't 100% solve the conflict. In the end the user must manually chose which version to keep, and which version to discard.

- If two identical lines exists in both versions but the lines is at another line number in the old document, this might cause a conflict <<< TEXT >> that could be avoided.

Alternatives considered: An algorithm that analyses every line in the file keeps the one that the user wants.

3.4 ER-Diagram

INSERT DIAGRAM HERE

3.5 User manual

3.5.1 Starting the application

- Running the application from Visual Studio
  Before starting the application, you need to start Visual Studio with administration priviledges. The reason for this is that the application will need to create files and folders for the documents and in order to do so the program needs to be run with administrator priviledges which will give the application write access.

  Since the web client runs in the browser, the browser needs to allow pop up windows for localhost. This can be done when the application is run for the first time.

  When starting the application, you need to set the WebClient project as startup project (if it's not already set), and then run the program (f5 for debug mode, ctrl + f5 for the release version).

  The web client will start up with your default internet browser. When the main page has loaded, you are ready to use the application.

- Signing up
  As a first time user there won't be any user registered in the system, so the first time you need to do is to sign up to use the system.

  To sign up, click the "Sign up" button. This will open up a new window with the sign up form.

  Fill out the form and click the "Sign up" button. A message will appear to say if the signup was successfull or not.

  If the signup was successfull you are now registered in the system, and are ready to use it. Close the sign up window and go back to the main window.

- Logging in
  On the main screen there are two text boxes at the top of the window named "username" and "password". Enter the newly created username and password into the boxes and click the "Login" button.

3.5.2 Using the Application

After you have logged in, press the "Get Files" button on the left page of the window. This will show you all the files that belong to the current user. Since you are a new user you don't have any documents, so you should only see the root folder (the one with your username).

- Create a new document.
  To create a new document, click the "New Document". This will clear all text boxes, and you are ready to write a new document.

  Creating a new document doesn't save the document, so before you go too far you should save your document. Write a filename in the filename box, and click the "Save Document" button. If you wish to save the document in a sub folder, just write the: "foldername/filename.html" in the filebox.

  The system doesn't require that you save the document as a HTML file, but the system is built around it. Not doing so won't make it able to add images to you document.

- Deleting a document
  Deleting a document is very simple. Select a document from the list on the left. Make sure that the filename of the file is entered into the filename box (this can also be done manually). Delete the document by clicking the "Delete Document" button.

- Sharing a document
  Sharing a document is very simple as well. Open the document you wish to share. Enter the username of the user you wish to share the document with in the text box next to the "Share Document" button, and click the share document button.

- Showing a document
  Since the document is built around the HTML format, the text area can't show any images or text formatting. In order to see the document (with images, formatting etc) you need to open it in another page in the browser. Select the document you wish to view and click the "Show

Document" button. This will open a new window with your document in parsed HTML.

4 Implementation

5 Project Management