Discourse Analysis Group

John Carrasquillo,Oliver Nichols,Crystal Ray,Heather Scott

CS 491

September 30, 2013

Table of Contents:

Current Progress page 3
Tasks to Complete page 4
Screenshots page 6

Current Progress:

- Overcame difficulties in retrieving and transferring the project’s repository and documentation over from the last semester’s group
- Generated documentation for setting up the project for the group that inherits the project next semester
- Debugged and corrected the previous group's PHP code and corrected the bug causing the upload process to throw an error
- Integrated the parser into the Website using PHP, thus essentially making it part of the Website itself
- Integrated the different pieces of the project into a single cohesive project
- Connected the applet from Fall 2012 due to issues with the Spring 2013 applet
- Implemented functionality to incorporate the client's desire to be able to parse unformatted text
- Implemented functionality to upload files and retrieve files from the database
- Updated website to display the user’s files stored in the database
- Updated database schema to follow a consistent naming convention

Tasks

Applet

Our Recommendation

We highly recommend NOT using the java applet. This creates additional work, but there are issues with using the java applet. A java applet cannot run in Chrome, it is seen as a high security risk in Firefox, and applet use has to be enabled in the browser for the applet to work at all. This limits the number of users that can or will use the software. The applet jar file certificate will also have to be resigned every 6 months to be usable in browsers. This will be cumbersome for maintenance after the project has been implemented and might be a problem for the client.

One recommendation is using JQuery and JQuery UI. Here is a link to JQuery UI where you can view some of its functionality: http://jqueryui.com/.

JQuery is JavaScript made easy. Essentially it is set up like an API in that you can reference the JQuery functions by including a reference to Google's hosted JQuery library. An example of this is as follows:

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js" ></script>
```

Note that the "1.10.2" might be an outdated library version, which would mean you would have to lookup the current library version and put its version number in place of the "1.10.2" in the example above.

Another option is to download it and host the library locally, but we recommend you use the Google-hosted library, which can be viewed at: http://code.google.com/apis/ajaxlibs/

It can easily manipulate HTML. It might be possible to replicate current applet functionality by allowing JQuery to use div, textarea, and similar HTML tags.

Tasks we noticed that are needed in the Java Applet from Spring 2013

1. Fix grouping functionality
2. Optimize merge functionality (smaller verse number is displayed, larger verse displayed in text)
3. Applet Split (conjunction highlighting)
4. The applet should not ask for a local file. It should use the file that it was given.

Website

1. fileUpload.php does not give feedback on upload errors to the user. It might be useful to return the user to upload.php, if the upload is unsuccessful and tell them the issue. The name field is required, but the user can upload a blank filename in some browsers. This bug should be fixed.

2. The website needs to be further connected to the applet (if it is continued). The applet should be passed the contents of the chosen file (using fileModule.php to get contents of file).
3. The website should display public files in addition to user files when the user views files. (There is a function in fileModule.php that can get all the public file information, it just needs to be placed into the website).
4. The registration input fields need to be required so that the user cannot leave them blank when submitting the form.
5. Query error in AdminModule.php from Spring 2013 that we found. (around line 305 and on)
6. Admin Options in the website need to be fully and correctly implemented.
7. (Suggestion) - If the user selects multiple files to view or edit, maybe open each file in a new tab.

Parser

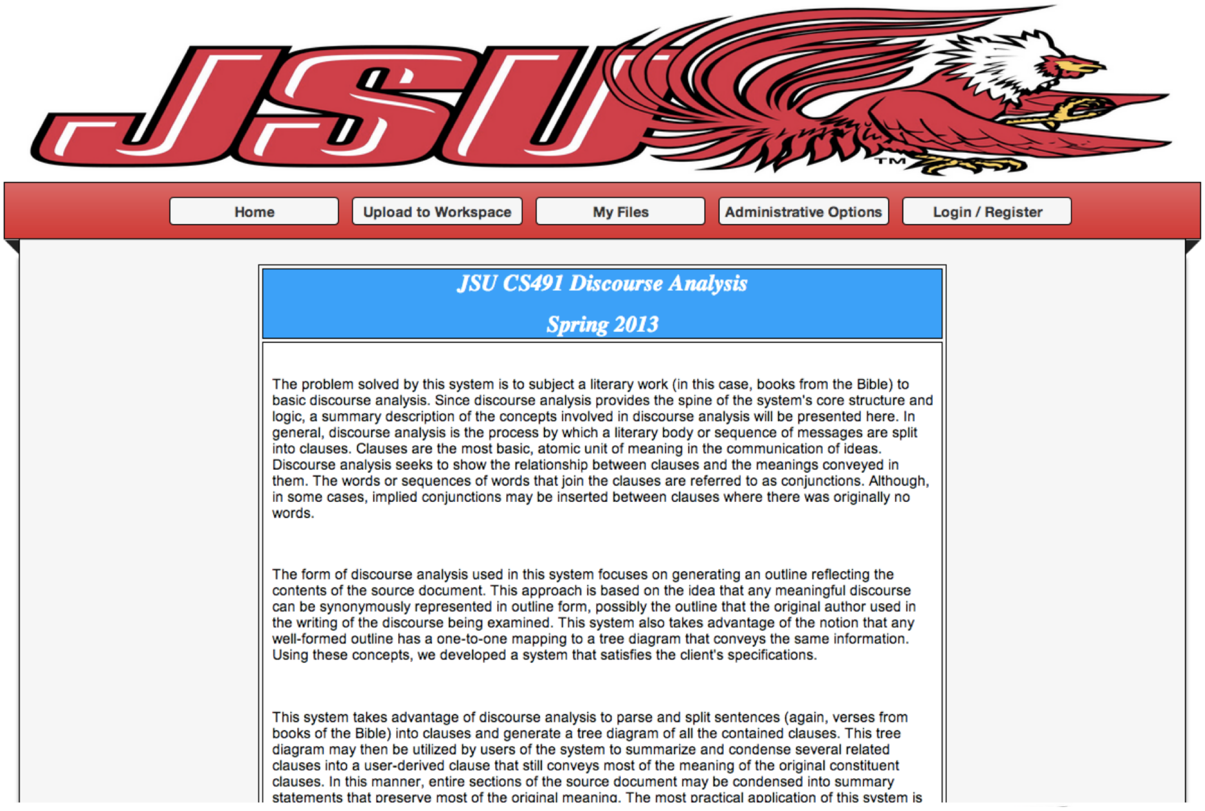
1. If there are any issues with parsing formatted text, first check that the text file was formatted correctly and then analyze the parser. The parser is in parser.php. When we tested the parser with correct formatting, the parser performed correctly.

Documentation

Be sure to document and add comments to new code so that future semesters can easily pick up where you left off.

Screenshots

Home



JSU CS491 Discourse Analysis

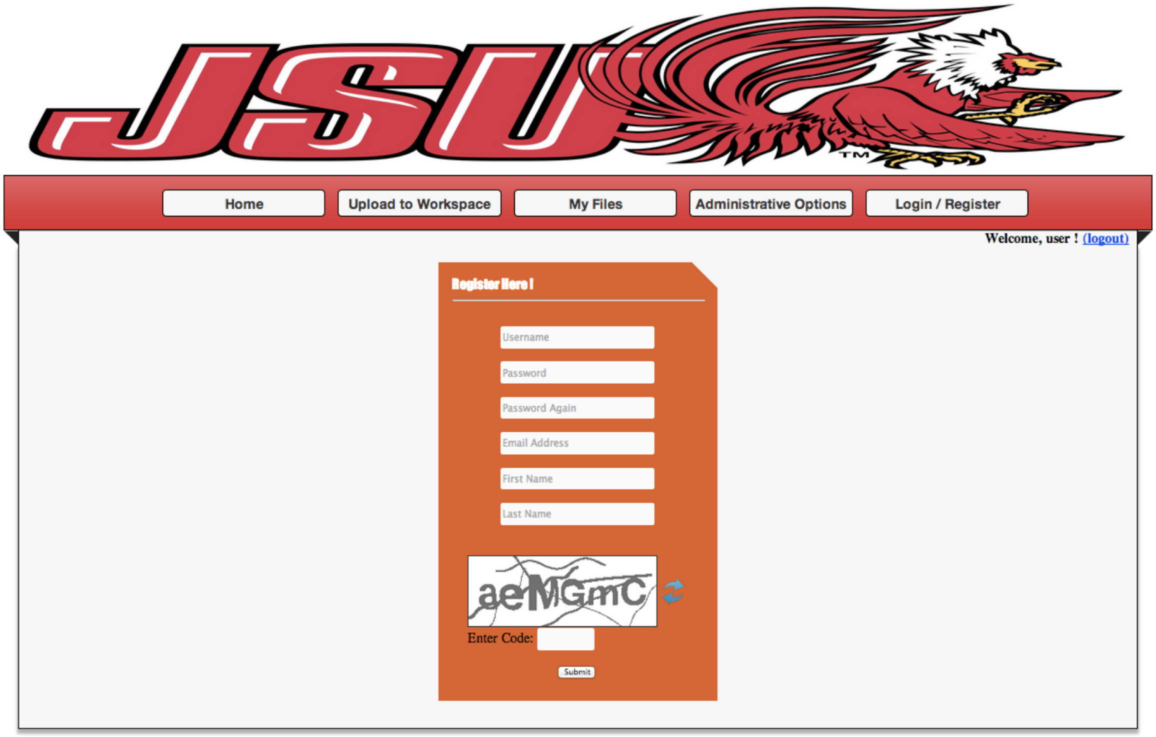
Spring 2013

The problem solved by this system is to subject a literary work (in this case, books from the Bible) to basic discourse analysis. Since discourse analysis provides the spine of the system's core structure and logic, a summary description of the concepts involved in discourse analysis will be presented here. In general, discourse analysis is the process by which a literary body or sequence of messages are split into clauses. Clauses are the most basic, atomic unit of meaning in the communication of ideas. Discourse analysis seeks to show the relationship between clauses and the meanings conveyed in them. The words or sequences of words that join the clauses are referred to as conjunctions. Although, in some cases, implied conjunctions may be inserted between clauses where there was originally no words.

The form of discourse analysis used in this system focuses on generating an outline reflecting the contents of the source document. This approach is based on the idea that any meaningful discourse can be synonymously represented in outline form, possibly the outline that the original author used in the writing of the discourse being examined. This system also takes advantage of the notion that any well-formed outline has a one-to-one mapping to a tree diagram that conveys the same information. Using these concepts, we developed a system that satisfies the client's specifications.

This system takes advantage of discourse analysis to parse and split sentences (again, verses from books of the Bible) into clauses and generate a tree diagram of all the contained clauses. This tree diagram may then be utilized by users of the system to summarize and condense several related clauses into a user-derived clause that still conveys most of the meaning of the original constituent clauses. In this manner, entire sections of the source document may be condensed into summary statements that preserve most of the original meaning. The most practical application of this system is

Registration



Welcome, user ! (logout)

Register Here!

Username

Password

Password Again

Email Address

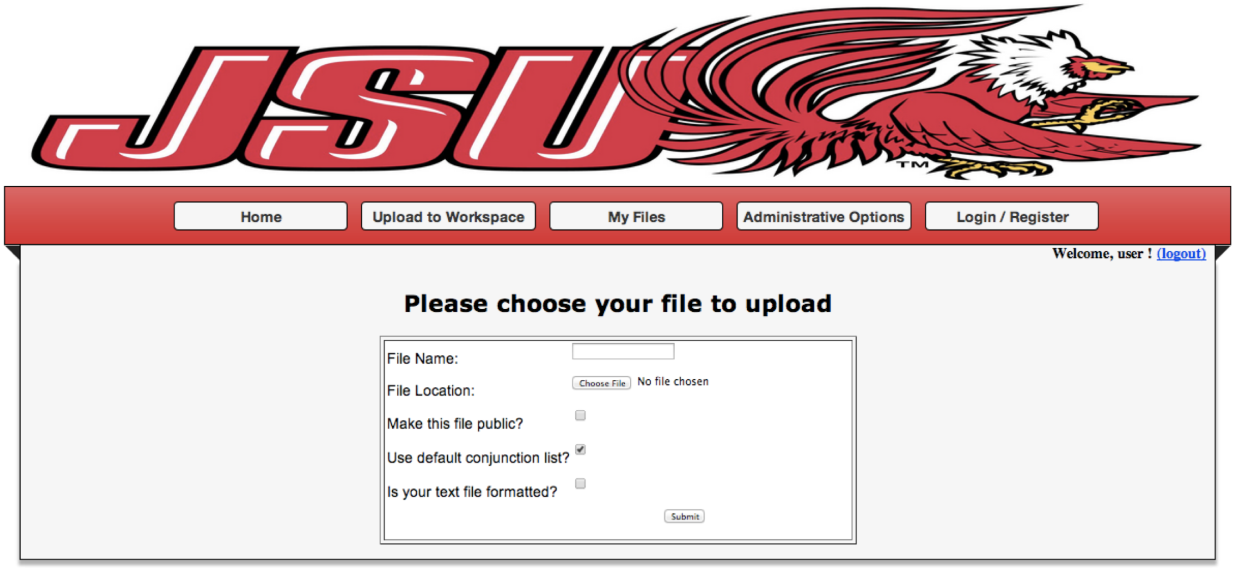
First Name

Last Name

aeMGmC

Enter Code:

Upload to Workspace



Welcome, user ! (logout)

Please choose your file to upload

File Name:

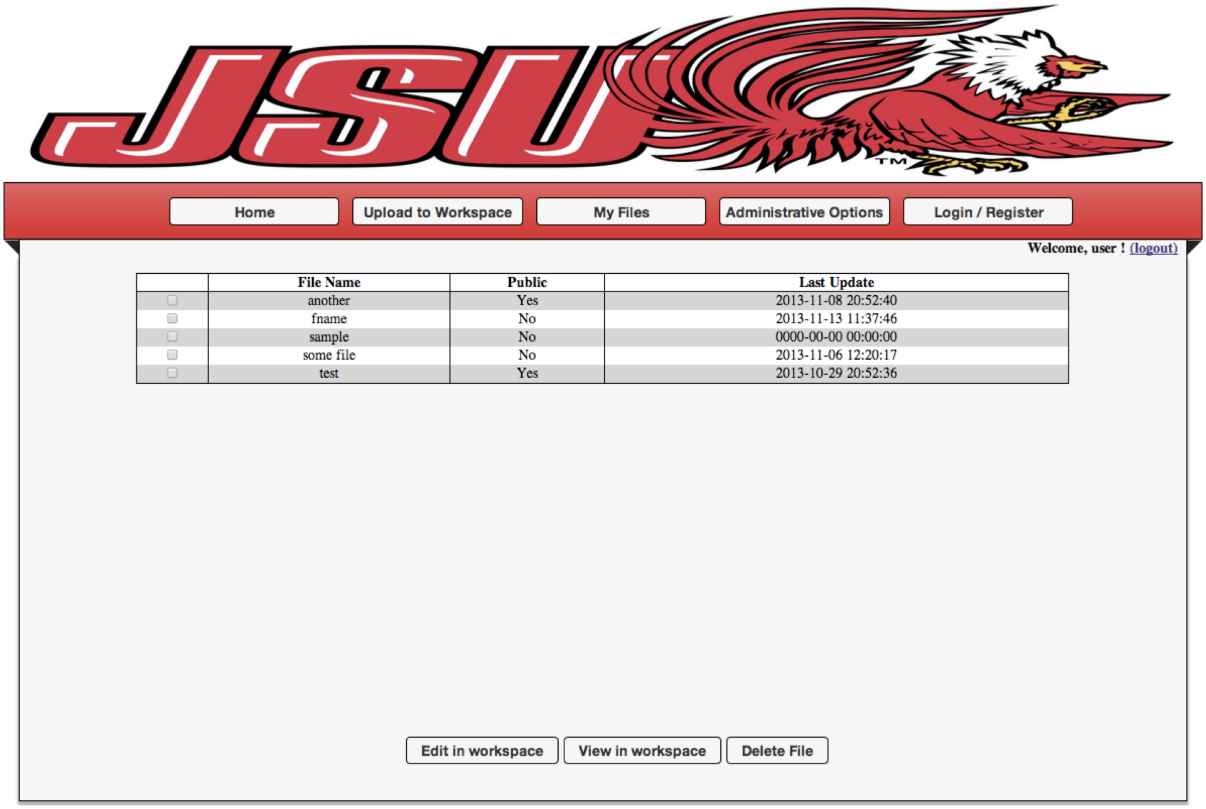
File Location: Choose File No file chosen

Make this file public? □

Use default conjunction list? ☑

Is your text file formatted? □

My Files



	File Name	Public	Last Update
☐	another	Yes	2013-11-08 20:52:40
☐	fname	No	2013-11-13 11:37:46
☐	sample	No	0000-00-00 00:00:00
☐	some file	No	2013-11-06 12:20:17
☐	test	Yes	2013-10-29 20:52:36