.GOVernator: Toward a Social Software Tool for Assessing Perspectives on Government Documents

By

Erhardt Graeff

Thesis submitted in partial fulfillment of the requirements for the degree of Honors Bachelor of Science in Information Technology

Rochester Institute of Technology

B. Thomas Golisano College
of
Computing and Information Sciences

May 26, 2006

Acknowledgments

First and foremost, I would like to thank my advisor Professor Tona Henderson for her support and motivation throughout this endeavor. This document would not be possible without her inspiration and encouragement.

I would also like to thank Professor Daniel Bogaard for his aid and advice in navigating a number of the programming problems I encountered during work on this project.

Lastly, I would to thank Professor Steven Zilora for his unyielding advocacy and concern for my completion of this thesis and graduation from the RIT Honors Program.

Abstract

This undergraduate Honors capstone project involves the creation of the novel web application called .GOVernator. The purpose of this social software tool is to allow users to "markup" government documents, like the *Bill of Rights*, with XML tags using an interface that does require in-depth knowledge of XML. The application is programmed using the Ruby on Rails framework with JavaScript and its implementation in Asynchronous JavaScript and XML (AJAX), XHTML, CSS, XML, and XSLT. The resulting program is prototype for allowing users to create an account via registration, navigate their own home page allowing them to select government documents to markup (a.k.a scrutinize), and then use a browser-based interface for adding XML tags to government documents—storing tag names to a database for later analysis. All program functionality, except for the critical markup functionality, is in place. Help documents are also still needed to guide the user through interaction with the application's interface. The future of the project is further development (programming) of the .GOVernator web application, a case study involving 10-30 users, and a proposal to the Lab for Social Computing as an adopted project for school year following this publication of this paper.

Table of Contents

Introduction

In this Honors Capstone Project my hope was to build an online social software application that acted as an ontological abstraction by using XML markup from user-tagged, publicly available government documents, accessed and manipulated through a browser-based application and stored on a server database. Accomplishing this lofty goal, in full, originally required three research components: a web application prototype, a case study, and a written proposal. .GOVernator, the browser-based application, is an newly authored, proprietarily programmed social software application, allowing multiple user accounts and one test government document (*The Bill of Rights*). The case study represents the analysis of would-be data from preliminary users of the system, instrumental in assessing the .GOVernator application as proof of concept. Finally, the written proposal would/will be a formal argument for the adoption of the .GOVernator project targeted at RIT's Lab for Social Computing (LSC). The following documentation refers exclusively to the first component—the prototype.

Background

I began working on a web application for marking up documents with XML tags during the summer following the completion of two grad courses regarding XML during the 2004-2005 school year at RIT. My interest in user-tagging (folksonomies) and the resultant XML documents and their semantics led me to investigate a social software tool that could facilitate such a process for non-code-savvy netizens. The original tool, called "the commentator" lacked a defining focus. Thus, after casually noticing that the average citizen lacked a general literacy in government documents, I thought that this common literacy and the perceptions of our crucial laws and defining documents could be analyzed through the folksonomies generated by the XML markup method. This is the idea behind .GOVernator.

In general, the .GOVernator application allows users to markup (or tag) a government document. Government documents were perfect fodder for a markup social software tool prototype for two major reasons: 1) they are free and publicly accessible and 2) there is, usually, some potential for debate or controversy over a specific point of legislation or significant clause. The hope, regarding the latter of these reasons, is that it will spur use of the system; eventual functionality may include the ability of users to see what everyone else (not credited to maintain anonymity) used on the documents as further encouragement. As previously mentioned, the positive civic outcomes derived

from .GOVernator would be to promote literacy of the legal documents governing our country as well as to potentially provide a novel source of understanding the public perception of the documents through tagging.

The application works for users by allowing them to add XML tags around the sections of text marking their personal reactions (the hope being that the process will glean semantically personal reflections on the content). Existing documents (already in the system) only have a base XML tag of <government_document>. All additional tags must be created using a proper XML tag name devised by the users, and checked for compatibility by the application. The actual tagging mechanism is designed so that users simply highlight a string of content and then type a word or underscored phrase into a form field to act as the name of the tag they wish to add—limiting the computer programming literacy needed to successfully manipulate the code by masking the underlying XML. All such data is saved as a new XML document to a database table and every entity tag will be saved to a dictionary table to facilitate easier analysis later. The XML documents are retrievable by their authors for re-inspection, but the database's "dictionary" table of used tag names will persist, in order to track every used idea for the specific content.

.Governator allows a potentially unlimited number of user accounts, requiring only a small amount of personal information to be stored (username, password, e-mail address, gender, and education level) for password retrieval and data analysis. The data analysis involved will be part of the proposed case study, using the "dictionary" and its reference source documents to compile a crude ontology of the language, content, and

perspectives on both government documents in general as well as specific works such as the *Bill of Rights*. The ontology would be capable of eventually facilitating the generation of general schemas for government documents based on statistically significant tags used in the system (including a mapping of synonyms) as well as specific schemas for each document. Documents can be later added after deployment, even per user request, to supply new fodder for criticism.

Project

The .GOVernator programming project is a merging of several cutting-edge and emerging (as of May 2006) web technologies in order to create a web application with a dynamic and easy-to-use interface. The main application structure is founded on the Ruby on Rails web application framework. Most backend functionality is programmed using the family of Ruby on Rails languages. Asynchronous JavaScript and XML (AJAX) is also a primary component, enabling user-manipulation of the source government documents in a non-refreshing browser window. XHTML, CSS, XML, and XSLT were also used during development. The database backend is MySQL—used to save all tagged documents and user data. Lastly, program testing is possible through the provided Ruby on Rails test server called WEBrick with the output tested in the Mozilla Firefox 1.5.0.3 browser.

Database Design

Before the programmatic elements are discussed in details, it is necessary to understand the underlying database structure on which the application is built. Five tables are in the database: users, documents, used_documents, tags, used_tags (see Appendix A for an Entity/Relationship diagram of the database). “Users” holds all necessary authentication and survey information taken from the user during registration and saved

in session variables after login. “Documents” contains the original XML markup with only root tags of the government documents that can be accessed and tagged by users. “Used_documents” then holds the newly tag government documents with reference to the User for ownership and the original source Document for title. The “tags” table keeps a running dictionary of all words or phrases used as tags by all users of the social software tool. “Used_tags” then maps specific words or phrases to a specific Used_document and the owner of that document/tag.

Application Components

.GOVernator comprises three major functional spaces allotted separate controller classes within the Ruby on Rails framework: user authentication system, user home page, and document markup. Each of these spaces will defined by their primary functions and interactive elements using a series of descriptions and screenshots throughout the remainder of this chapter.

User Authentication System

The user authentication system takes care of new user registration (see Figure 1) as well as login of previously registered users (see Figure 2).

Figure 1: .GOVernator's registration page

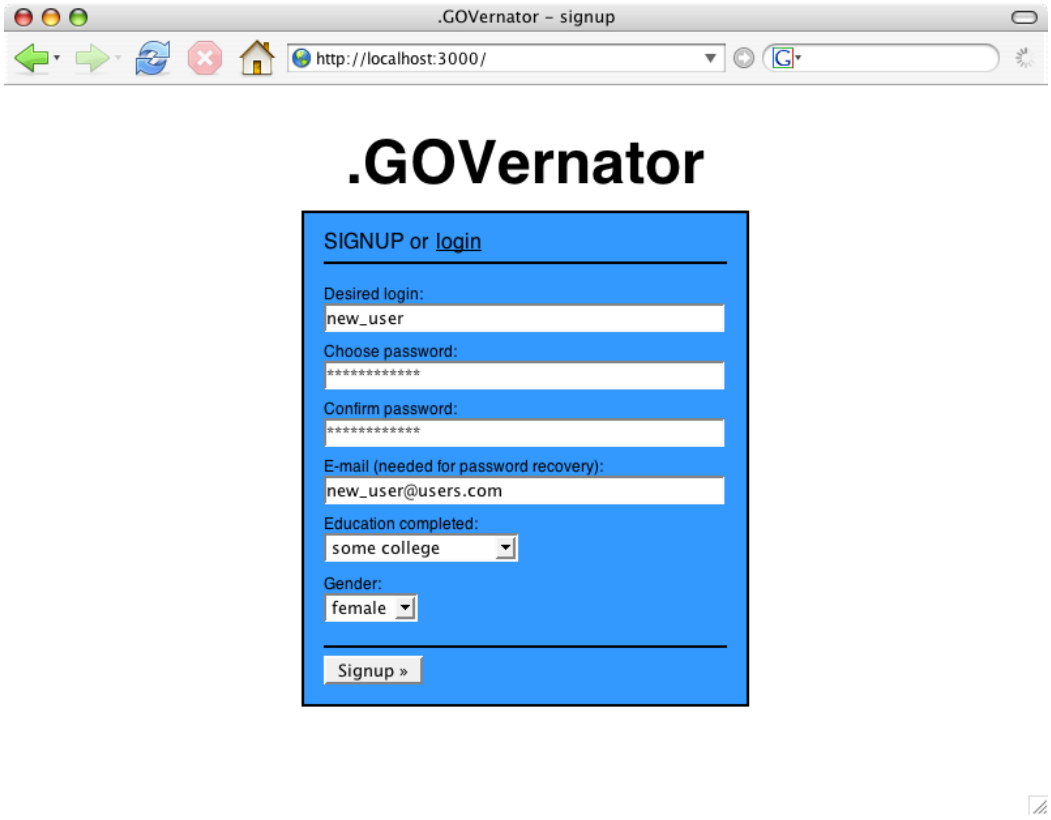


Figure 2: .GOVernator's login page

User Home Page

The home page for each user is displayed after a successful registration or login for that user. Several options are provided for the user at this stage. S/he can: logout, markup (scrutinize) a new government document, or edit a previously "scrutinized" document. The home page display is adjusted based on which documents have been scrutinized by the user already (see Figures 3, 4, and 5).

Figure 3: .GOVernator's user home page display (no documents have been scrutinized).

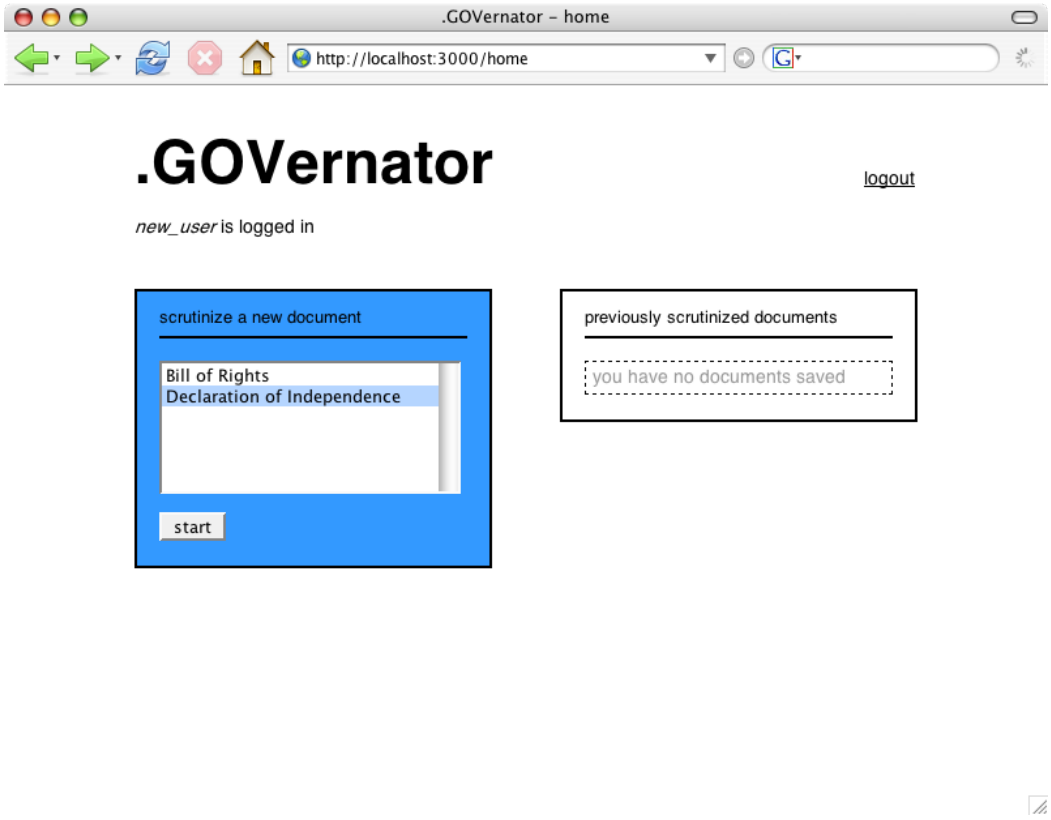


Figure 4: .GOVernator's user home page display (one document has been scrutinized).

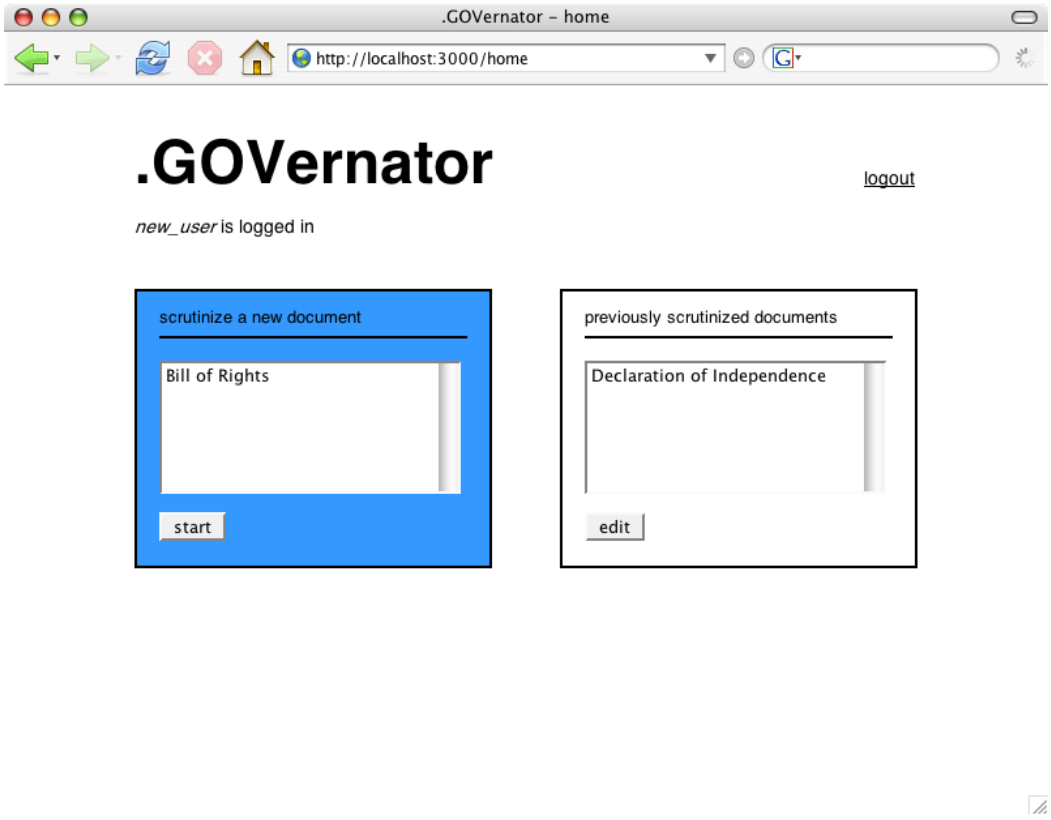
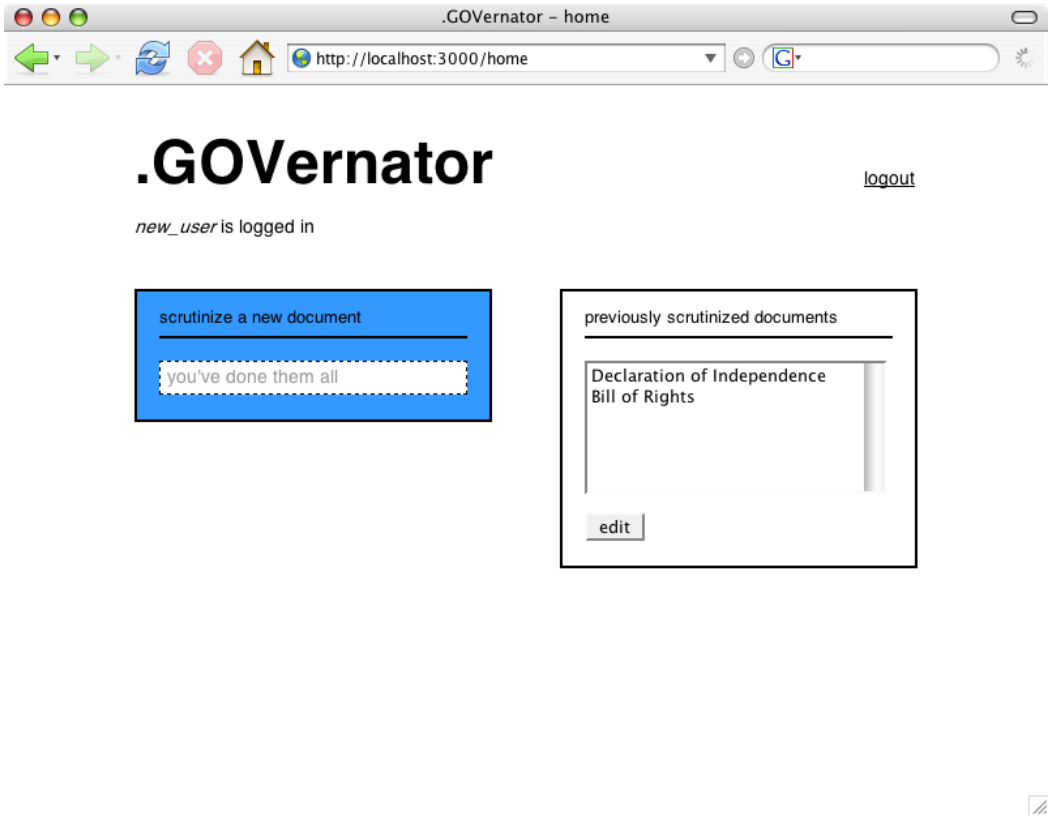


Figure 5: .GOVernator's user home page display (all documents have been scrutinized).



Document Markup

In this last major function space or component of .GOVernator is where the marking up of government documents actually takes place. The interface displays the XML version of the government document selected by the user (see Figure 6). The user is then allowed to selected portions of the document by highlighting them with the cursor. By performing this action, a JavaScript function is called to reveal a hidden div containing a form displaying the selected text as well as an input box for the user to enter a word or underscored phrase as the XML tag to enclose the selected text (see Figure 7). Once submitted, the tag is added to the database table Used_tags after checking for its

existence in the Tags table (and either adding the word or phrase or referencing it). The entire enclosed selection is also added to the Used_tags table into the "context" field. The display "should" then update through AJAX to reflect the changes made in the markup. Once the user is finished marking up his or her document, they can click the "save and exit" button to return to their home page.

Figure 6: .GOVernator's markup page displaying a portion of the *Bill of Rights*.

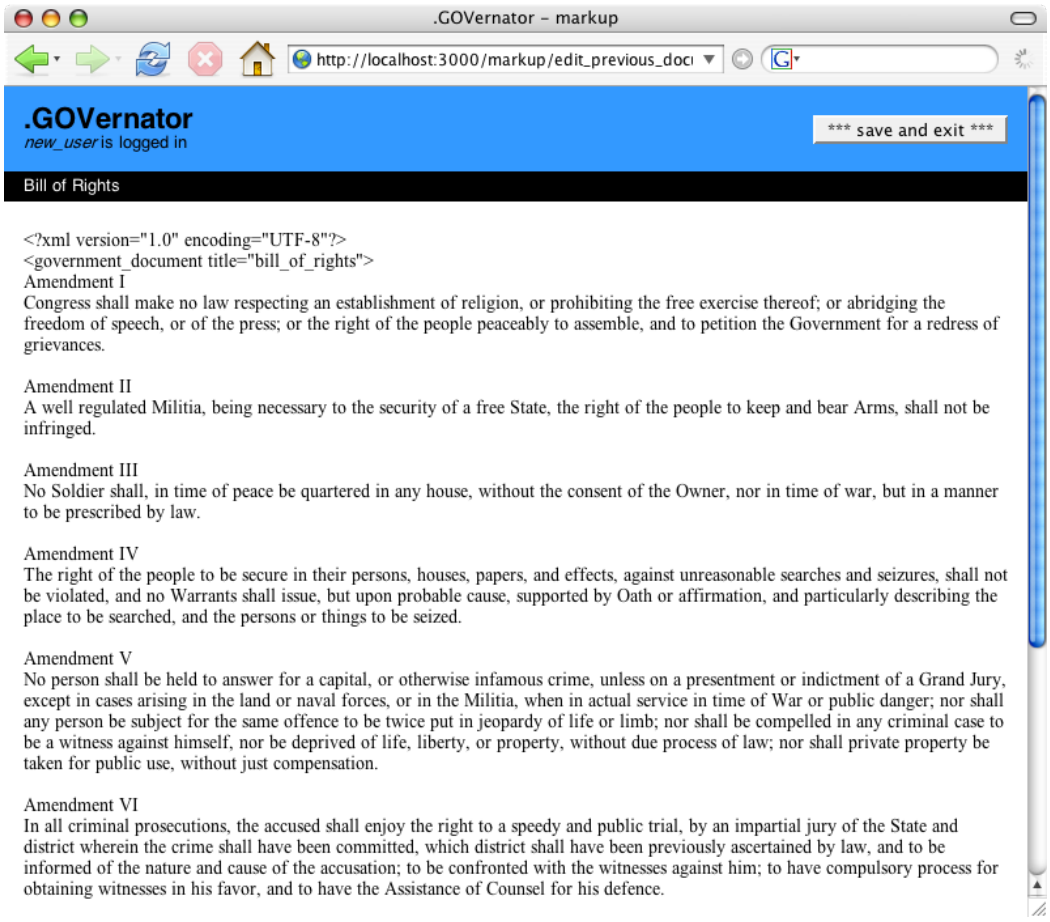
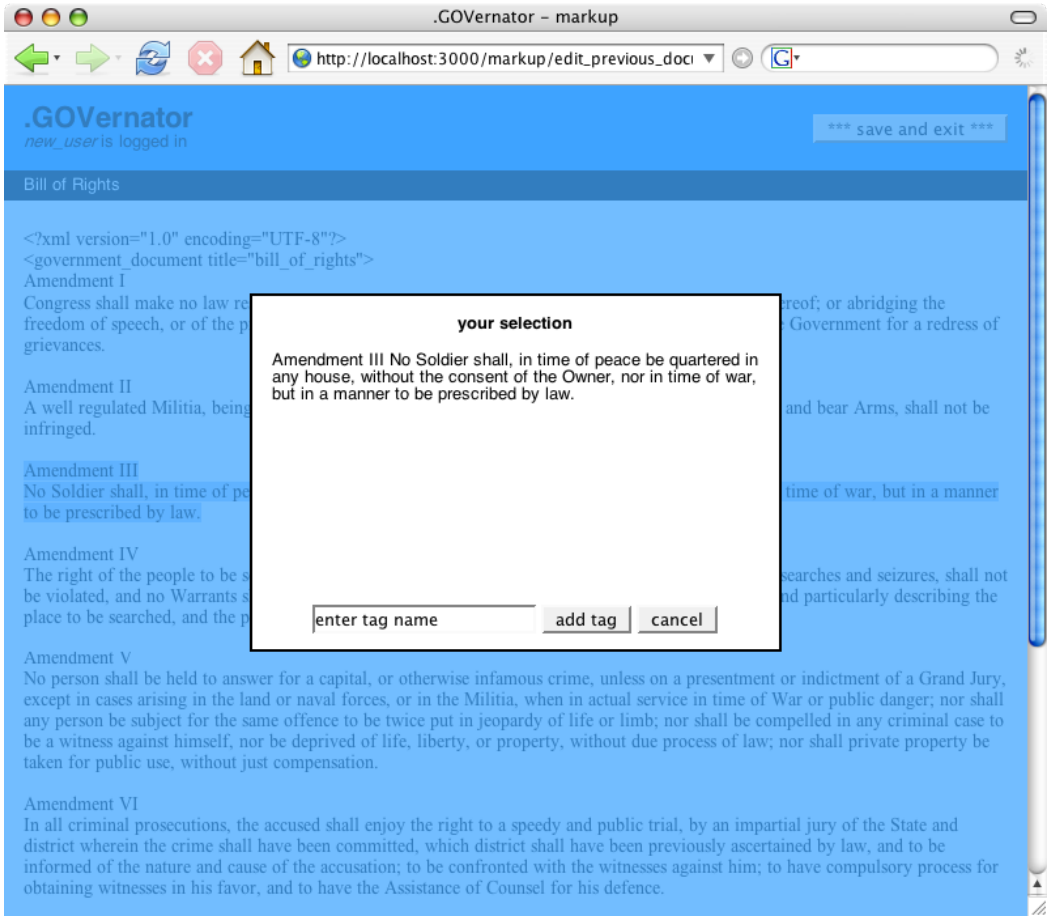


Figure 7: .GOVernator's markup page with the revealed form after highlighting.



Results

By the end of the allotted time for development, the social software tool prototype of .GOVernator failed to accomplish all programmatic and functional goals set in the original design. While basic user and database access was intact and fully operational, the web application lacked critical functionality in regard to the main objective of "marking-up" official government documents.

The most basic level of programmatic success was in the application running as intended without error. Most of the component successes involved foundational elements of the .GOVernator application that were devised, installed or built, configured, and tested during the creation of the development environment. This includes proper database design as defined in the previous chapter, connection to the database and proper action and page routing with the testing server (Ruby on Rails WEBrick server) running.

As evidenced by the screenshots provided in the previous chapter, the user login system was comprehensively successful in regard to its intended functionality. The system could register new users with required survey and authentication information. Users were then effectively logged into the system after registration. Once logged in, users were capable of navigation a "home" page in order to either logout or markup documents. The status of documents owned by the user was managed correctly by the

application and interface. And the logging out functionality was in place—with the system capable of logging in previously registered users, thereafter.

The critical functionality of allowing users to markup the available documents (*Declaration of Independence* and *The Bill of Rights* used in testing) is incomplete. First, the display of the XML versions of available government documents had to be simplified, due to the inability of Ruby on Rails or JavaScript to transform the XML using XSLT and then add the transformed document tree to the necessary element on the "Markup" page. This issue resulted in a less-than-ideal string substitution method handled through Ruby programming that failed to have the intended formatting and coloration effects (see Appendix B for example *Bill of Rights* XML document, Appendix C for intended XSLT document, and Appendix D for the intended transformation output).

The JavaScript-powered interface allowed for highlighting passages and inputting desired XML tag names into a selection-specific form. However, the AJAX to Ruby on Rails calls failed to save the newly tagged string of desired text to the database. This missing piece *is* the proposed ability of users' to markup documents using .GOVernator. Thus, the prototype fails to support the defining, critical function of the application.

Recommendations

.GOVernator is incomplete. Thus, the following list of recommendations all stem for the central and paramount recommendation: complete the endeavor. Firstly, the needed programmatic functionality of the web application should be finished in a way that allows for a proof of concept assessment. Secondly, the original plan to carryout a case study should be pursued. Thirdly, a proposal to the Lab for Social Computing should be drafted based on the case study results. And lastly, the social software application should be published online through either the LSC or another venue, so that general internet users can use the tool.

Missing functionality in the current state of the .GOVernator application was outlined in the preceding chapter. All enumerated issues require a significant amount of programming to overcome. Incompatibility of the system across browsers is another potential issue that will be addressed after the completion of the original project as defined in the introduction chapter of this paper. The final functional component, thus far unmentioned yet necessary to the success of the proof of concept, is help documentation built into the application website, which explains how to use the tool's interface.

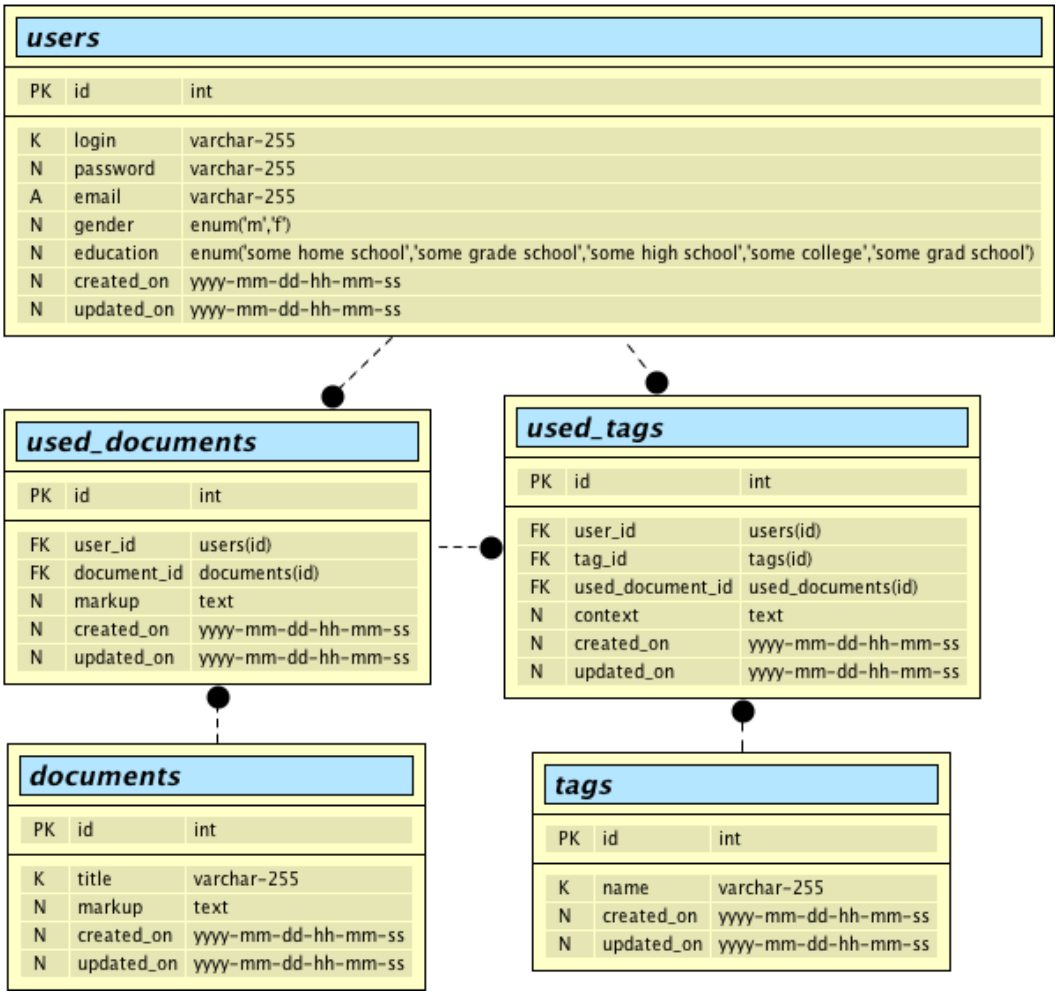
Once all required application functionality for the .GOVernator prototype is achieved, the case study should be conducted using a preliminary sample size of 10 users. This can later be expanded to a sample size of 30, an appropriately large sample for

statistics, if the 10 fails to any statistically normal characteristics in the distribution of chosen tag names. The used_tags table of the database will serve as the raw data for analysis. Statistics regarding the choice of a specific word, as an XML tag, will be mapped against its relative position in the source documents as well as the survey information collected about the user(s) that employed it.

After the case study can be completed, data analyzed, and statistics compiled, the proposal to the LSC should be written. The proposal should address the necessary concerns of the Lab in order to make an effective case for adoption of the .GOVernator project as an official project for the next school year (beginning September 2006). I intend to work at the LSC throughout next year (September 2006 – May 2007), while I finish a second degree in International Studies—this would allow me to be the principle investigator of the project on behalf of the Lab.

The end goal of the development phase and final recommendation for .GOVernator is deployment of the application online. I believe that there is a substantial desire for this social software tool and for the civic discourse that could be facilitated by it. I have, also, no doubt in my mind that RIT should be the host of such a novel interconnection of information technology and civics.

Appendix A:
Entity/Relationship Diagram of the Database



Appendix B:
Example *Bill of Rights* XML Document

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="Bill_of_Rights.xsl" ?>
<government_document title="bill_of_rights">
    Amendment I
    Congress shall make no law respecting an establishment of religion,
or prohibiting the free exercise thereof; or abridging the freedom of
speech, or of the press; or the right of the people peaceably to
assemble, and to petition the Government for a redress of grievances.

    Amendment II
    A well regulated Militia, being necessary to the security of a free
State, the right of the people to keep and bear Arms, shall not be
infringed.

    Amendment III
    No Soldier shall, in time of peace be quartered in any house,
without the consent of the Owner, nor in time of war, but in a manner
to be prescribed by law.

    Amendment IV
    The right of the people to be secure in their persons, houses,
papers, and effects, against unreasonable searches and seizures, shall
not be violated, and no Warrants shall issue, but upon probable cause,
supported by Oath or affirmation, and particularly describing the place
to be searched, and the persons or things to be seized.

    Amendment V
    No person <lame_amendment>shall be held to answer for a capital, or
otherwise infamous crime, unless on a presentment or indictment of a
Grand Jury, except in cases arising in the land or naval forces, or in
the Militia, when in actual service in time of War or public danger;
nor shall any person be subject for the same offence to be twice put in
jeopardy of life or limb; nor shall be compelled in any criminal case
to be a witness against himself, nor be deprived of life, liberty, or
```

property, without due process of law; nor shall private property be taken for public use, without just compensation.

Amendment VI

In all criminal prosecutions, the accused shall enjoy the right to a speedy and public trial, by an impartial jury of the State and district wherein the crime shall have been committed, which district shall have been previously ascertained by law, and to be informed of the nature and cause of the accusation; to be confronted with the witnesses against him; to have compulsory process for obtaining witnesses in his favor, and to have the Assistance of Counsel for his defence.

Amendment VII

In Suits at common law, where the value in controversy shall exceed twenty dollars, the right of trial by jury shall be preserved, and no fact tried by a jury, shall be otherwise re-examined in any Court of the United States, than according to the rules of the common law.

Amendment VIII

Excessive bail shall not be required, nor excessive fines imposed, nor cruel and unusual punishments inflicted.

Amendment IX

The enumeration in the Constitution, of certain rights, shall</lame_amendment> not be construed to deny or disparage others retained by the people.

Amendment X

The powers not delegated to the United States by the Constitution, nor prohibited by it to the States, are reserved to the States respectively, or to the people.
</government_document>

Appendix C:
XSLT Document (excluding xml-to-string by Evan Lenz)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
    <xsl:import href="xml-to-string.xsl"/>

    <xsl:output method="html" indent="yes" version="1.0" omit-xml-
declaration="yes" media-type="text/html" encoding="UTF-8"/>

    <xsl:template match="/">
        <xsl:apply-templates/>
    </xsl:template>

    <xsl:template match="government_document/child::node()">
        <xsl:variable name="all_text">
            <xsl:call-template name="xml-to-string"/>
        </xsl:variable>

        <xsl:call-template name="lf2br">
            <xsl:with-param name="all_text" select="$all_text"/>
        </xsl:call-template>
    </xsl:template>

    <xsl:template name="lf2br">
        <xsl:param name="all_text"/>

        <xsl:variable name="lf">
            <xsl:text>
</xsl:text>
        </xsl:variable>

        <xsl:choose>
            <!-- When a linefeed is found, process it correctly -->
            <xsl:when test="contains($all_text,$lf)">
                <xsl:variable name="text_before_lf" select="substring-
before($all_text,$lf)"/>
```

```
                <xsl:variable name="text_after_lf" select="substring-
after($all_text, $lf)"/>

                <!-- If tags are in text before linefeed, then
Highlight the XML Tags in Red -->
                <xsl:if test="contains($text_before_lf,'&lt;')">
                    <xsl:value-of select="substring-
before($text_before_lf,'&lt;')"/>

                    <xsl:variable name="tag_text" select="substring-
after($text_before_lf,'&lt;')"/>

                    <span style="color:#f33;">
                        <xsl:text>&lt;</xsl:text><xsl:value-of
select="substring-before($tag_text,'&gt;')"/><xsl:text>&gt;</xsl:text>
                    </span>

                    <xsl:value-of select="substring-
after($text_before_lf,'&gt;')"/>
                </xsl:if>

                <!-- If no tags, then just output text before the
linefeed -->
                <xsl:if test="not (contains($text_before_lf,'&lt;'))">
                    <xsl:value-of select="$text_before_lf"/>
                </xsl:if>

                <!-- Add newline for lf character -->
                <br/>

                <!-- Go back to beginning of template function with
just remaining text -->
                <xsl:call-template name="lf2br">
                    <xsl:with-param name="all_text">
                        <xsl:value-of select="$text_after_lf"/>
                    </xsl:with-param>
                </xsl:call-template>
            </xsl:when>

            <!-- When tags are found before a linefeed, highlight them
and then cycle back through template to catch the actual linefeed -->
            <xsl:when test="contains($all_text,'&lt;')">
                <xsl:value-of select="substring-
before($all_text,'&lt;')"/>
```

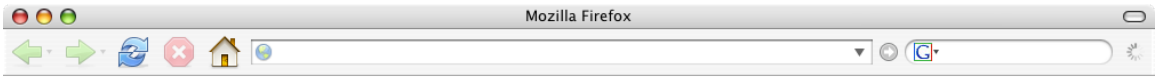
```
                <xsl:variable name="tag_text" select="substring-
after($all_text,'&lt;')"/>

                <span style="color:#f33;">
                    <xsl:text>&lt;</xsl:text><xsl:value-of
select="substring-before($tag_text,'&gt;')"/><xsl:text>&gt;</xsl:text>
                </span>

                <xsl:call-template name="lf2br">
                    <xsl:with-param name="all_text">
                        <xsl:value-of select="substring-
after($all_text,'&gt;')"/>
                    </xsl:with-param>
                </xsl:call-template>
            </xsl:when>

            <xsl:otherwise>
                <xsl:value-of select="$all_text"/>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>
</xsl:stylesheet>
```

Appendix D:
Proposed XSL Transformation Output (tags in red)



Amendment I
Congress shall make no law respecting an establishment of religion, or prohibiting the free exercise thereof; or abridging the freedom of speech, or of the press; or the right of the people peaceably to assemble, and to petition the Government for a redress of grievances.

Amendment II
A well regulated Militia, being necessary to the security of a free State, the right of the people to keep and bear Arms, shall not be infringed.

Amendment III
No Soldier shall, in time of peace be quartered in any house, without the consent of the Owner, nor in time of war, but in a manner to be prescribed by law.

Amendment IV
The right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures, shall not be violated, and no Warrants shall issue, but upon probable cause, supported by Oath or affirmation, and particularly describing the place to be searched, and the persons or things to be seized.

Amendment V
No person <lame_amendment>shall be held to answer for a capital, or otherwise infamous crime, unless on a presentment or indictment of a Grand Jury, except in cases arising in the land or naval forces, or in the Militia, when in actual service in time of War or public danger; nor shall any person be subject for the same offence to be twice put in jeopardy of life or limb; nor shall be compelled in any criminal case to be a witness against himself, nor be deprived of life, liberty, or property, without due process of law; nor shall private property be taken for public use, without just compensation.

Amendment VI
In all criminal prosecutions, the accused shall enjoy the right to a speedy and public trial, by an impartial jury of the State and district wherein the crime shall have been committed, which district shall have been previously ascertained by law, and to be informed of the nature and cause of the accusation; to be confronted with the witnesses against him; to have compulsory process for obtaining witnesses in his favor, and to have the Assistance of Counsel for his defence.

Amendment VII
In Suits at common law, where the value in controversy shall exceed twenty dollars, the right of trial by jury shall be preserved, and no fact tried by a jury, shall be otherwise re-examined in any Court of the United States, than according to the rules of the common law.

Amendment VIII
Excessive bail shall not be required, nor excessive fines imposed, nor cruel and unusual punishments inflicted.

Amendment IX
The enumeration in the Constitution, of certain rights, shall</lame_amendment> not be construed to deny or disparage others retained by the people.

Amendment X
The powers not delegated to the United States by the Constitution, nor prohibited by it to the States, are reserved to the States respectively, or to the people.