DropFiles.com – Business to Business File Sharing

By

Timothy J Boland

Submitted to
the Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Computer Science Technology

University of Cincinnati
College of Applied Science

May 2005

DropFiles.com – Business to Business File Sharing

by

Timothy J Boland

Submitted to
the Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements
for
the Degree of Bachelor of Science
in Computer ScienceTechnology

© Copyright 2005 Timothy J Boland

The author grants to the Information Engineering Technology Program permission
to reproduce and distribute copies of this document in whole or in part.

_____	_____
Timothy J Boland	Date
_____	_____
Professor Sanyal	Date
_____	_____
Patrick C Kumpf, Ed.D. Interim Department Head	Date

Acknowledgement

I would like to thank Professor Sanyal for guiding me along this project timeline. We held weekly meetings during Senior Design II that helped me keep on track. Second I would like to give special thank my fiancé Amanda Smith for her support and patience during throughout this project.

Table of Contents

Abstract

File sharing between businesses has become an important technical challenge facing IT interims of security, reliability and simplicity. Commons solutions include FTP, e-mail, and WebDAV have their limitations and security concerns. DropFiles.com is an answer to many of these security problems, combined with a straightforward interface and process flow. No longer is there a need for IT intervention or hardware to help coordinate transferring large files to another company. Users only need web access, a DropFiles account, and the ability to browse to their file for an automated process to track, secure, and maintain uploaded files.

Introduction

I have observed in work and college experience that one constant problem has been the inability to get large files from network to network. At work I am constantly being asked if I can put a large file on a Web server for a client or vendor to download because of the file attachment limitation of e-mail servers. These files include Photoshop files, TIFF's, PDF, JAR, and word documents among others. The company has a FTP server, but because of the complications and lack of security in FTP, this solution is rarely used. Trying to explain FTP to non-technical users can also be challenging. Sending an HTTP hyperlink in an e-mail is more user friendly. All the user has to do is simply click on the link and the download process begins. This application can automate the process for businesses to share files with clients and vendors in a fast, simple, and secure solution without involving their information technology department. This application is also exposed as a Web service to allow other types of collaborating software to consume this functionality. The primary emphasis of this project is database and web application programming.

This proposal outlines the current problem of sharing large files between businesses in detail, along with my solution. Also included are the design protocols, user profiles, deliverables of the project, proof of design, and a conclusion and recommendations section. Some of the industries this application is designed for include marketing agencies, digital photographers, multimedia production agencies, and outsourcing companies. This application could also be utilized for personal and student usage. Another target group of this application is remote workers. With business starting to adopt Virtual Private Networks, getting files off a private network can be slow in speed

because of the encryption of the data which slows the speed of a download. The Web service aspect of the application allows other software and Web applications the ability to have secure file sharing functionality. Some software that this could be utilized are instant messaging, wiki software, and bulletin board forums.

1.0 Statement of the Problem

Transferring large files between non-technical users across networks is an increasing problem today because there are more and more remote users and users on the go. One solution for this is to use a FTP server. However, FTP by its nature is insecure because of clear text passwords being sent across network lines. If a malicious user wanted to gain access to a FTP server he/she could packet sniff the network and see the unencrypted password (1). For most organizations this is unacceptable because of confidential files residing on the FTP server. FTP servers are also unable to track if a file was downloaded or not. For non-technical users FTP can be very challenging to use even with a client interface as opposed to a Web browser.

Sending large files as e-mail attachments is usually not an option because most e-mail administrators limit the file size attachment. According to the California Institute of Technology most universities set their file attachment (incoming or outgoing) limitation to 10 megabytes (2). At the University of Cincinnati, the mail servers are limited to 7 megabytes of file transfer (3). Another problem with e-mailing files is that many virus scanning software block e-mails with certain file attachments such as .zip, or .exe. This leaves users unable to share large files securely and reliably.

A more recent solution for transferring files has been Web-based Distributed Authoring and Versioning (WebDAV). While WebDAV does indeed provide a solution

to securing files, it however lacks a user friendly graphical user interface and can much more complicated to end users then FTP.

2.0 Description of the Solution

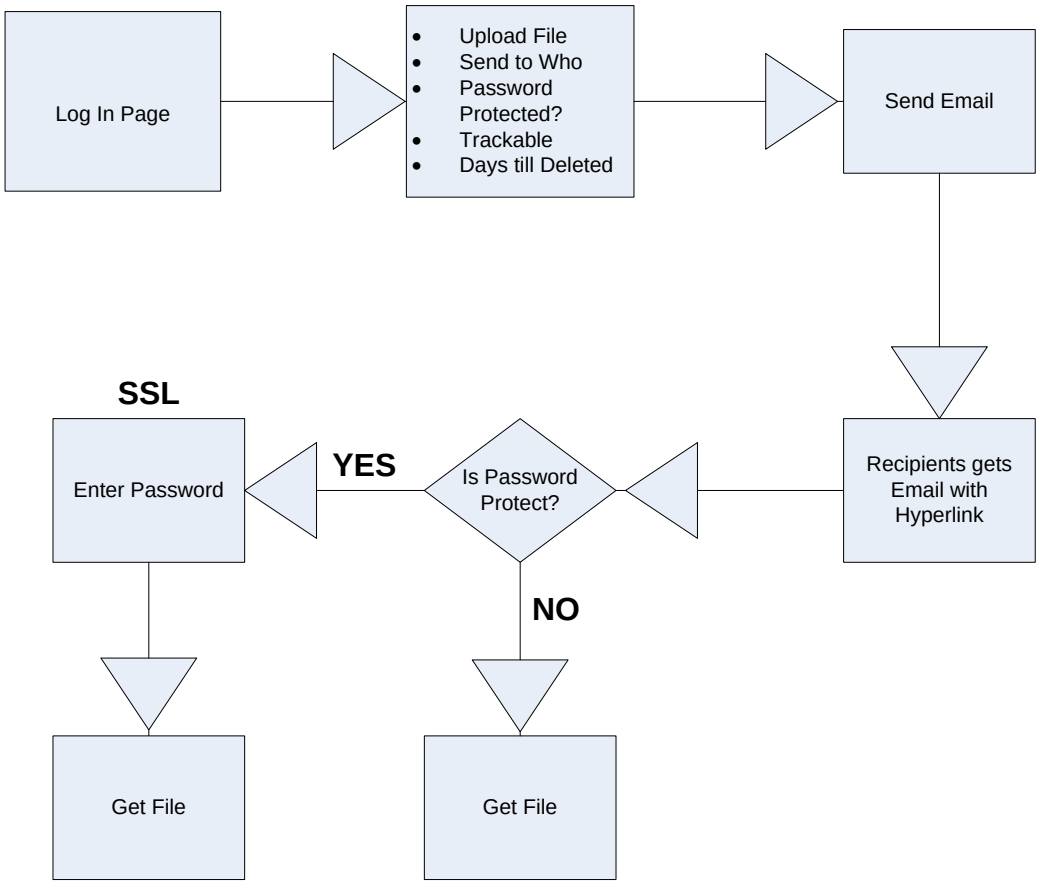
This application automates the process in which business exchange large files. The areas of focus of this application are web application programming and a relational database scheme. A major benefit of the application is to not only automate the process but to do so securely and to simplifying the process to the end users. The most complicated aspect of the project is to make the process as simple as possible.

Users of the system will visit DropFiles.com to upload files and enter in the recipients e-mail address to send them an e-mail with a hyperlink to pick up their files. Users of the system will have several options to track, choose how many days the file will live on the server, and secure the file over Secure Socket Layer (SSL) with a password. This application will also be exposed as a web service to allow other collaborating types of software or websites the ability to tap into the functionality of DropFiles.com.

DropFiles.com utilizes ASP.NET for the server side programming platform with VB.NET as the code behind language. Microsoft SQL 2000 is being used for the relational database. The following are important descriptions of the solution.

2.1 Design Protocols

The following diagram outlines the process flow of DropFiles.com.



Organizational Scheme

DropFiles is constructed using three primary areas of discipline, relational database scheme, web application programming, and multimedia. The relational database scheme section will layout the database in a relationship diagram, explanation of design and implementation. Actor analysis, use cases and a use case diagram will demonstrate the object oriented approach to the web application programming. The

multimedia section layouts the user interface design and techniques used for the look and feel of the website.

Database Design

The database has been defined into three objects for design. The customers, password protected files, and non-password protected files. This resulted into a six table design with one lookup table for the files. The primary key for the customers object is customerid which is a bigint data type. It is the foreign key in the following tables: user_login, NoPasswordFiles, PasswordFiles. The lookup table is used for fast indexing on files uploaded into the system and to determine if they are password protected or not. An index was built on the e-mail column in the user_login table for faster performance on user logon. Figure 1 is a diagram of the database.

Storing files in the database itself was not chosen for several reasons. For hard drive disk performance, there is an extra step that needs to occur when the file is retrieved from the database, this can affect scalability of the application. The other significant reason for not storing files in the database is that the application was not intended to be database specific. The database portion of the application could easily be ported to MySQL, Oracle or DB2 without major configuration changes.

By designing the database without being commercially database specific no stored procedures, views, or triggers are being utilized. Instead I am using server side coding to do all database calls and action.

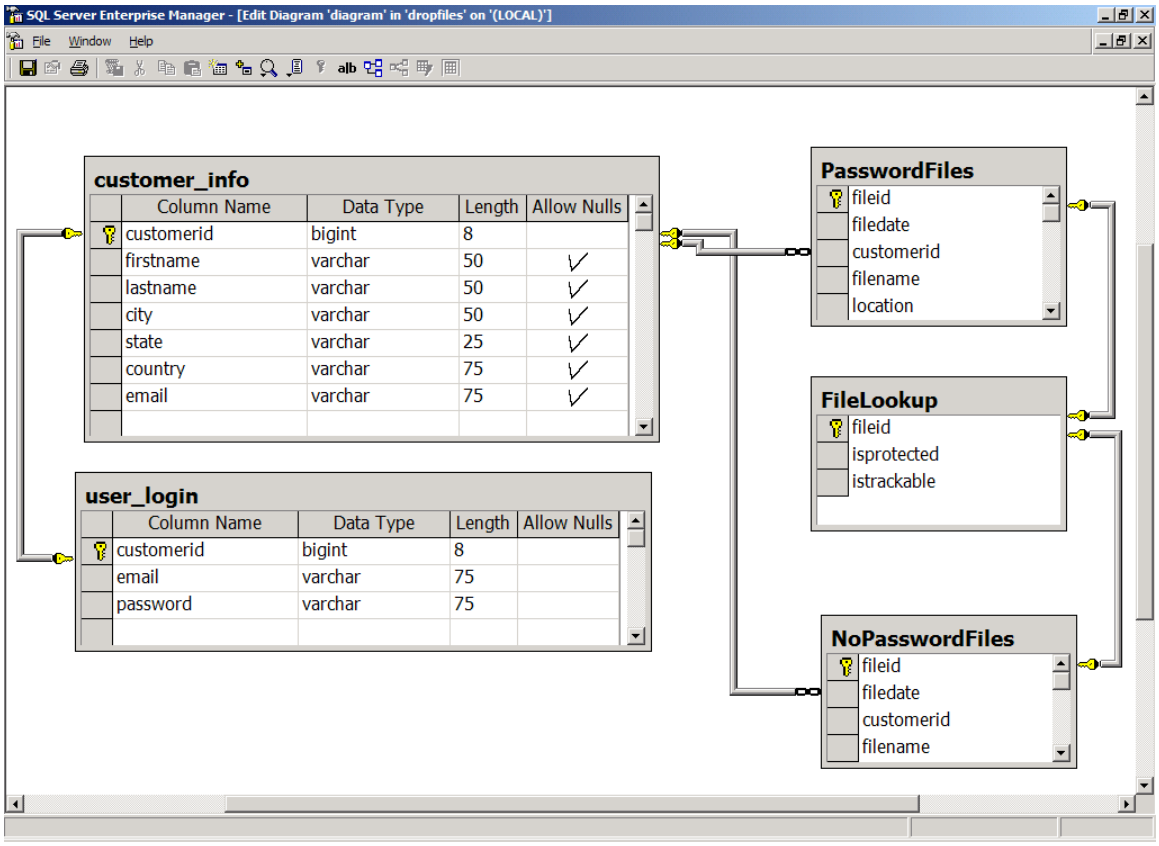


Figure 1. Database Relationship Diagram

Programming Structure

Object oriented approach is utilized in the web application. Each page has a code behind view that then calls various classes and their methods that are needed. Early binding of objects and their data types were used for increased performance in ASP.NET. The web.config file is being used to store the database connection string object and authorization to the applications itself. Storing the connection string in the web.config file allows it to be reference anywhere in the application without having to initialize it as a new class or object. Figure 2 demonstrates the usage of this setting. This file is protected by the .NET framework and can not be access through a web browser.

```
<appSettings>
    <add key="ConnectionString"
value="server=localhost;database=dropfiles;uid=dropuser;pas
sword=dropfiles4ever;" />
  </appSettings>
```

Figure 2. web.config connection string setting.

The ADO.NET Datareader object has been used wherever possible for the increased performance over the Dataset object. All passwords are stored using MD5 hashing algorithm for security. If the system would ever become compromised, the malicious users would not be able to read passwords in clear text. Web controls are used for doing server side includes for the navigation. The client side programming is done using HTML 4.01, CSS 2.0, and JavaScript. The JavaScript rollovers are written using custom attributes of html tags. On the body load the external JavaScript function is called that then parses the page looking for the custom attribute. Figure 3 highlights the custom attribute being used on dropfiles.com

```
<A href="products.html" ><IMG height=32 alt="" src="images/left_home.gif"
width=140 border=0 rollover="images/left_home_f2.gif" ></A>
```

Figure 3. Highlighted JavaScript custom attribute

User Interface

The user interface is designed with clear graphics, focused for the types of users of the system. The top navigation, left navigation, right news section, and bottom navigation are featured on every page of the site, giving it a consistent look throughout. Users are presented with 3 consistent paths to travel throughout the site each containing the same links with rollovers. The site is justified to the left site and has a width of 700 pixels, which fits into 800 x 600 screen resolution or higher. I have chosen to have a white background and with blue color for larger html fonts while graphics fonts including

the navigation use red color font. Figure 4 demonstrates the design of the site and the login page.

Figure 4. Website Design

HTTP Protocol

File uploading and downloading will occur over port 80 – HTTP. This will allow for cross platform client side support. By using HTTP, network firewall issue will be adverted as opposed to FTP which uses port 20 and 21.

SSL

Users will have the option to password protect files, allowing only those who know the password to download that particular file. Password authentication will occur over 128 bit encrypted SSL. The SSL was setup on IIS 5.5 using Microsoft's SSLSelf toolkit.

Tracking

Members will have the option to determine if a file was picked up. When a file that is labeled as trackable is downloaded, the users IP address and timestamp will be recorded. An additional email will be automatically sent to the member notify them that the file was downloaded.

Days till Deleted

Members of the system will be able to specify how long a file is to remain on the server or if it never should be deleted. A cron like job will be run at midnight on the server to delete files that have exceeded their time to live.

2.2 User Profile

There are three categories of users of DropFiles.com, paying members, non paying users, and recipients.

Paying Members

These are the primary users of DropFiles.com. Anyone can signup for this service, but specifically DropFiles.com is targeted at the following industries: marketing agencies, advertising agencies, digital photographers, multimedia production agencies, and outsourcing companies. These industries are constantly sharing assets between themselves on project. For instance, a marketing agency usually works with graphic shops sending back and forth large in size Photoshop files.

Users are comfortable with daily computer task and familiar with the World Wide Web and e-mail. Most have bachelor degrees in marketing, art, or business. These are non technical users who are intimidated with any non daily computer task such as using FTP.

Non Paying Users

These users have the same characteristics of a paying member but are given less functionality of DropFiles.com. They are only allowed to upload files, password protect those files and send to only paying members.

Recipients

This is the user who receives the e-mail from the paying members. These users could have a wide variety of background. This could be a client of the member, meaning a more business oriented person, or it could a member from a similar industry of the paying member.

2.2.1 Actor Analysis

The following chart identifies the actors and their specific goals for using the web application.

Actor	Goals

Paying Member	• Upload File
	  ○ Authenticate
	  ○ Set Password For File
	  ○ Set Days Till Deleted
	  ○ Set Files as Trackable
	  ○ Send E-mail to Recipient
	• Reporting
	  ○ See uploaded files and trackable file information

Non Paying User	• Upload File
	  ○ Send to only paying members
	  ○ Set Password For File

Recipient	• Pick up File ○ Authenticate for password protected files

2.2.2 Use Cases

The following use cases outline the three actors and their interaction with their primary goal from above.

Use Case Number: 1	UC1: Upload File
Last Updated:	01/25/05
Actor	Paying Member
Objectives	Enable paying member to upload file, specify password, days file stays on server, if the file is trackable, and recipients e-mail address.
Pre conditions	Paying member must have an active account, be within their file size limit for the total of their uploaded files.
Results: (Post-conditions):	File is uploaded to the web server, and an e-mail is sent to their recipient with a hyper link to pick up their file.
Actions 1. Member visits dropload.com/fileupload 3. Member enters username and password 5. Member uploads file, specifies recipients e-mail address, has the option to track the file and password protect the file.	**System Response** 2. System prompts for username and password 4. System validates password for the username, forwards to upload file page. 6. System puts file on web server. Writes file information to database, and send e-mail to recipients e-mail address.
Approved exceptions	

Use Case Number: 2	UC2: Upload File
Last Updated:	01/31/05
Actor	Non Paying User
Objectives	Enable non paying users to upload files, assign passwords and send to only paying members.
Pre conditions	Must be sending to paying member and know their e-mail address.
Results: (Post-conditions):	File is uploaded to the web server, and an e-mail is sent to paying member with a hyper link to pick up their file.
Actions 1. User visits dropload.com/sendafile 3. User enters members e-mail address 5. User uploads file, and has the option password protect the file.	**System Response** 2. System prompts for members e-mail address for verification. 4. System validates e-mail address forwards to upload file page. 6. System puts file on web server. Writes file information to database, and send e-mail to members e-mail address.
Approved exceptions	

Use Case Number: 3	UC3: Pick up File
Last Updated:	01/31/05
Actor	Recipient
Objectives	Enable recipient to download file.
Pre conditions	Recipient gets e-mail with hyperlink and know password if applicable.
Results: (Post-conditions):	File is downloaded to recipient's computer.
Actions 1. Recipient clicks link in their e-mail sent from dropfiles.com 3. Recipient enters password	**System Response** 2. System looks up file. If file is password protected, prompts user for password. Else skip to step 5 4. System verifies password. 5. System sends file to recipient's web browser. 6. If file is determined to be trackable, IP address is recorded, timestamp, and e-mail is sent to the member notify them of pickup.
Approved exceptions	**File has expired its time to live on server**

2.2.3 Use Case Diagram

Figure 5 shows the use case diagram for actors and their roles.

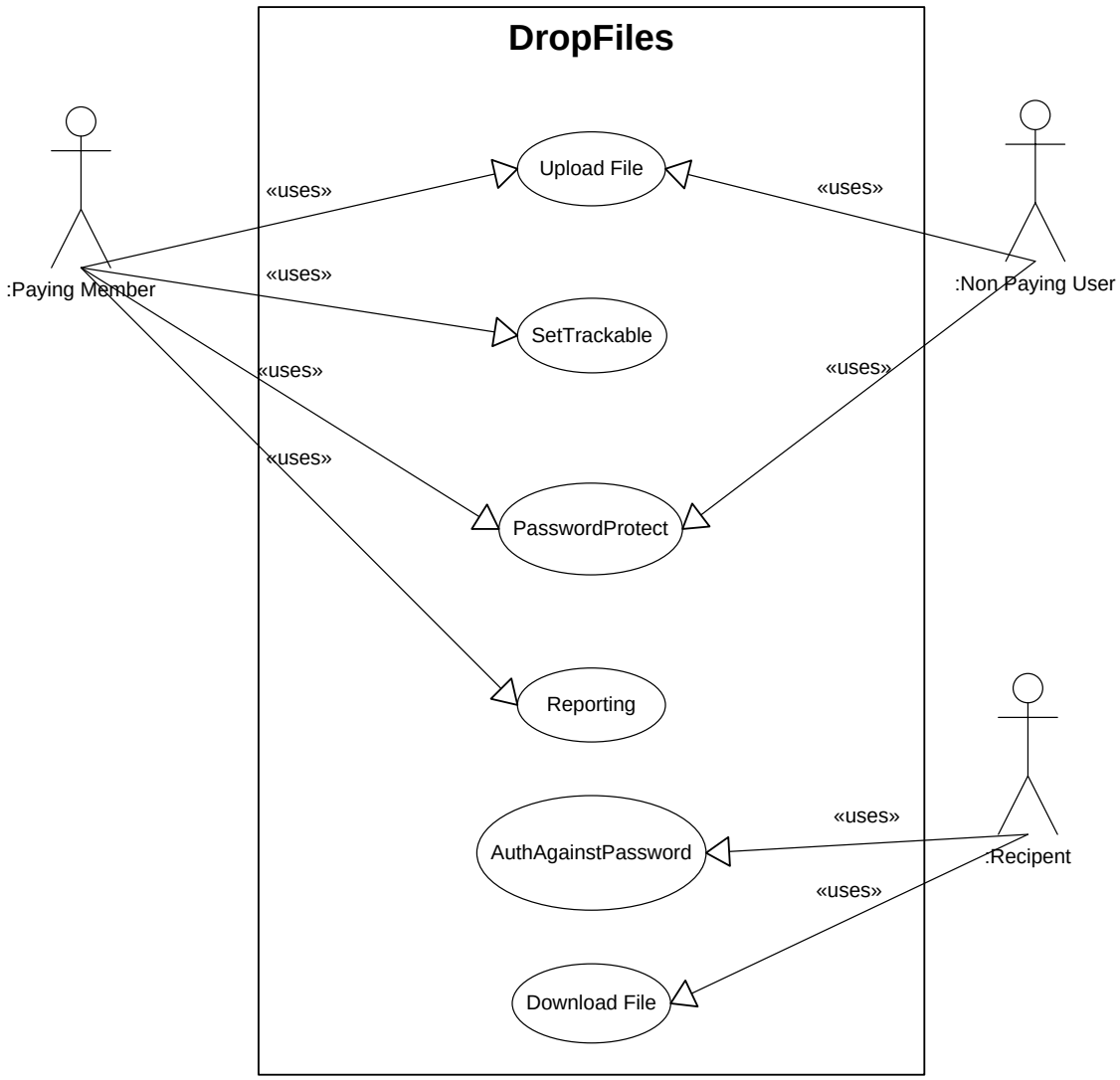


Figure 5. Use Case Diagram

3.0 Deliverables

The following are the final deliverables developed during the design freeze for dropfiles.com:

1. A Web application that allows business to business file sharing

2. A Web application developed in ASP.NET using Visual Basic.NET for the programming login
3. A relational Microsoft SQL 2000 server database
4. Enable users defined as members to do the following:
    a. Authenticate against the system
    b. Upload files up to 500 megabytes in size.
    c. Password protect uploaded file
    d. Set files as trackable
    e. E-mail recipient with hyperlink to download file
    f. Set how long a file remains on the server
    g. Access a management interface for reporting
5. Enable all other users to do the following:
    a. Gain access to send a file by identify a members e-mail address
    b. Upload file up to 500 megabytes in size
    c. Password protect uploaded file
    d. E-mail member only with hyperlink to download file
6. Enable recipient to do the following:
    a. Download file by clicking on hyperlink in e-mail
    b. Authenticate against password if one specified
7. Expose upload functionality as a web service Application Programming Interface (API).

4.0 Proof of Design

This section outlines in detail how all deliverables have been met for the project.

4.1 Authentication

Authentication for members access uses the FormsAuthentication method provided by the .NET framework. Using it in conjunction with settings in the web.config file I am able to say anyone accessing any page in the dropfile.com/fileupload directory must be authenticated and are redirect to the dropfiles.com/logon.aspx page. Passwords are stored in user_login table hashed, so I take what the member enters as their password, hash it, then do a compare, returning a value of 0 if they match. Validation for blank fields happen on both the client side, using JavaScript, and server side incase the user has JavaScript disabled. Figure 6 below demonstrates an incorrect password entered.

Figure 6. Incorrect Password Entered

Because I used the FormsAuthentication method I had to create and encrypt my own FormAuthentication ticket when a non member of the site was trying to send a file to a member.

I used an in between table (FileLookup table) in the database to allow for a quick lookup if a file was password protected and redirect them to a page to enter their password. In the FileLookup table I am able to add the fileid value to the recipients session to later validate the password with that associated fileid. All authentication happens over SSL for security reasons.

4.2 File Uploading and Receiving Files

All file uploading whether done by a member or a non member user of the system is handled by the FileUploading.vb class. By setting the html form attribute encType equal to "multipart/form-data" I am able to specify an input type equal to file (<input type=”file” name=”myfile”>). File attributes such as file size, file name, which virtual directory the file is stored, are written to either the NoPasswordFiles or PasswordFiles tables based on if a password was set. Every file that is uploaded has a 10 random alpha character appended to the name to prevent phising attacks. A member has options that a user sending to a member does not have. Tracking and Days till deleted are not shown for users sending to a member. They are accessing the same page, but through an object oriented design, these options are not shown. Figure 7 and Figure 8 below demonstrate the two different interfaces.

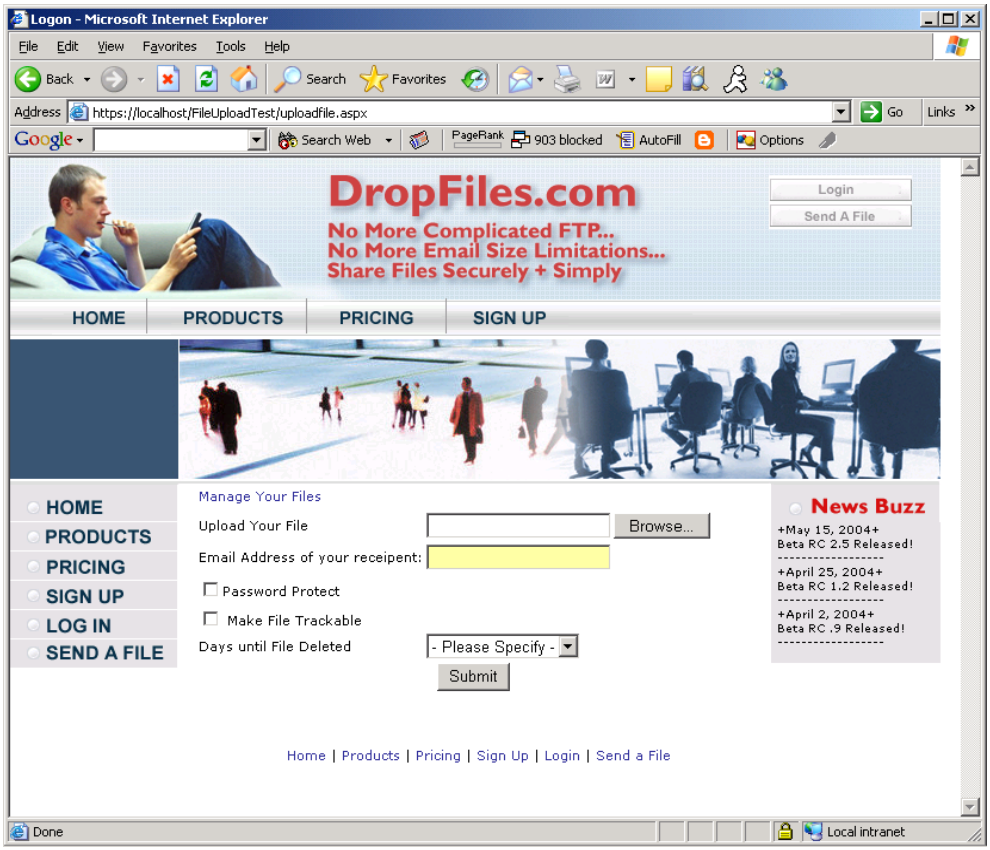


Figure 7. Member Interface Screen

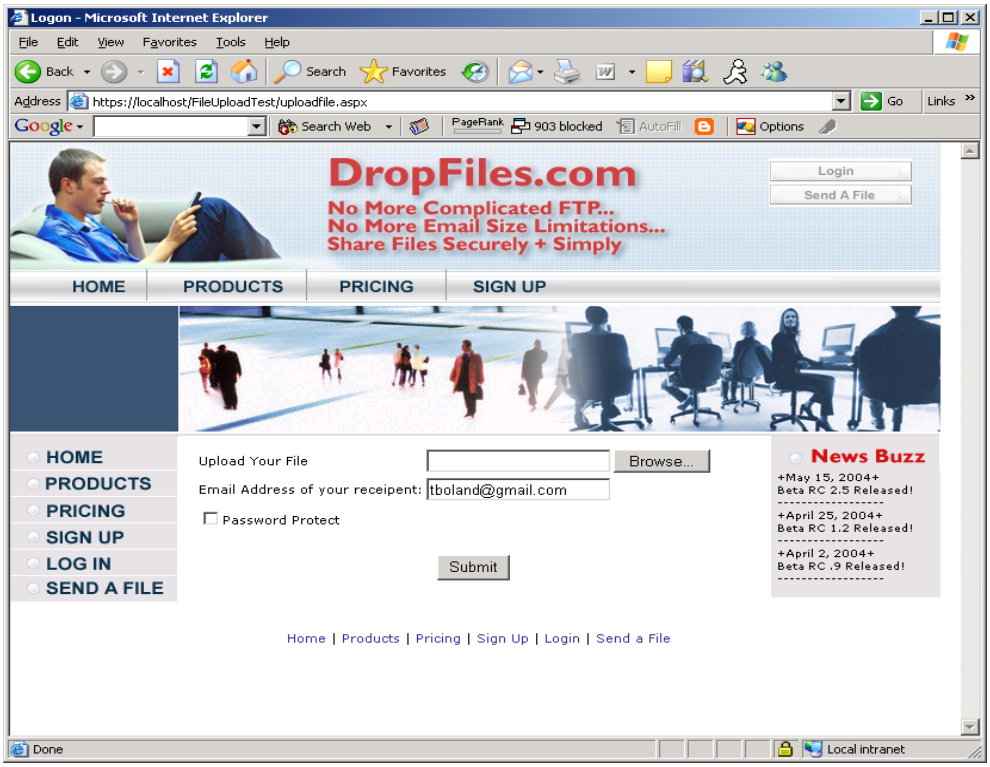


Figure 8. Interface for a User Sending a Member a File

A random 20 alpha characters are chosen for each file that is written to the FileLookup table and used as the query string variable when the file is picked up. For example https://dropfiles.com/pickup/pickup.aspx?f=qcuuqpvriqcnukeyxoiw the 20 random variable is after the f=. The recipient gets this URL in the email that is sent; notify them someone has uploaded a file for them to download. Figure 9 below shows the email the recipient receives.

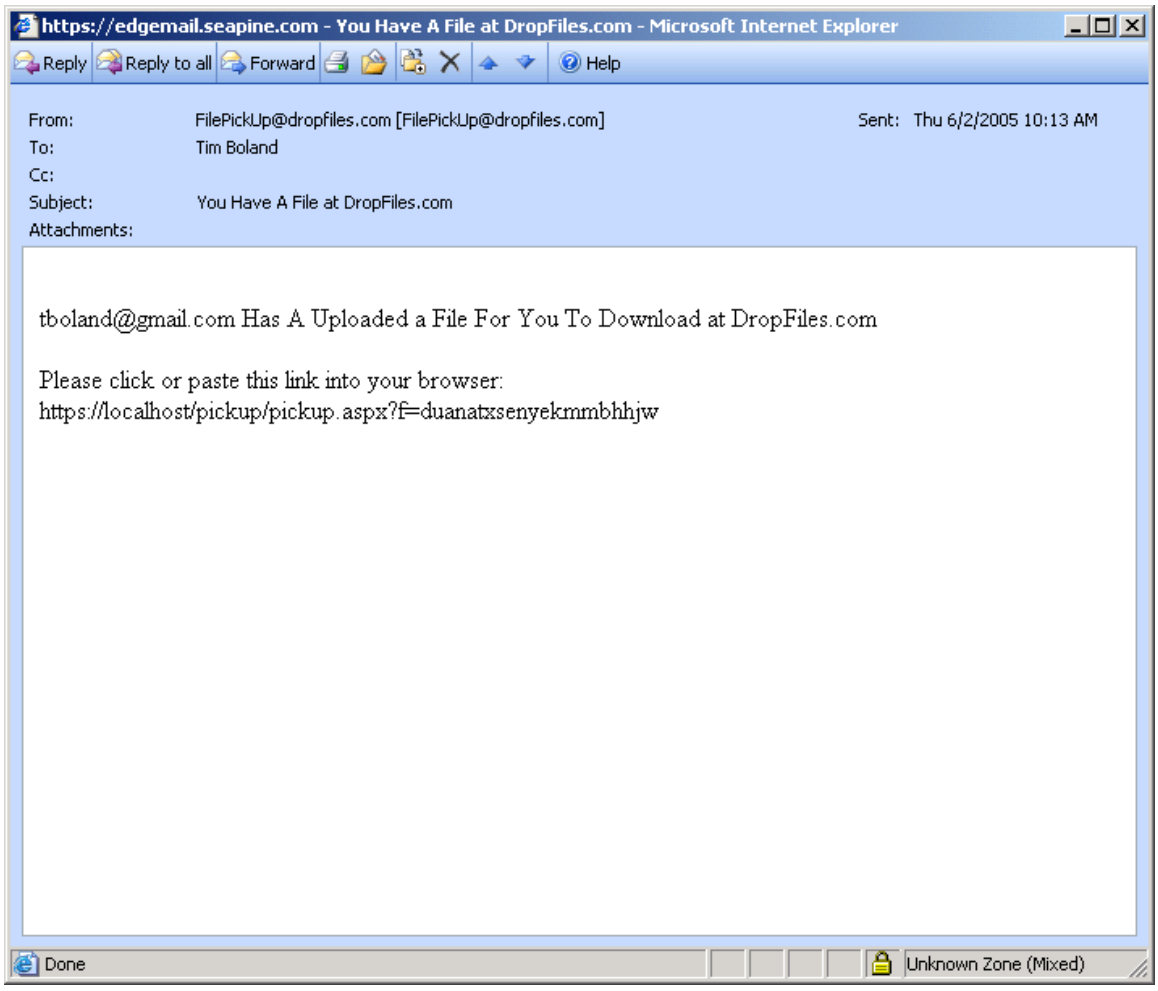


Figure 9. Email Recipient Receives

Picking up the file, the recipient either is directed to a enter a password over a SSL page or is streamed out their file if no password was specified. The figure below

demonstrates a password entered into the system and file being streamed back to the web browser.

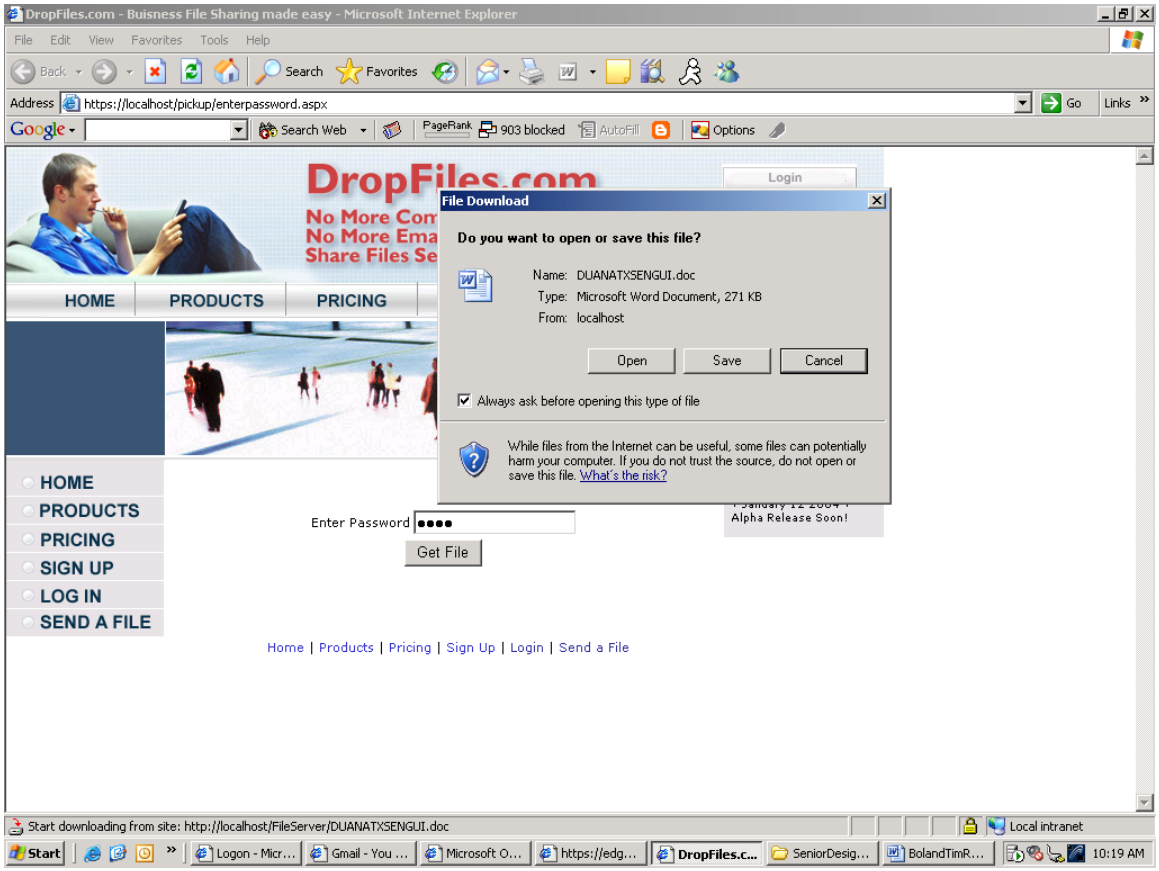


Figure 10. Password Verified and File Sent to Browser

4.3 Tracking and Days Till Deleted

If a file is set to be tracked, when a recipient picks up that file, an email is sent to the member notifying them that file was picked up, what time, and what IP address the recipient was coming from. Below is a screen shot of the email containing this information.

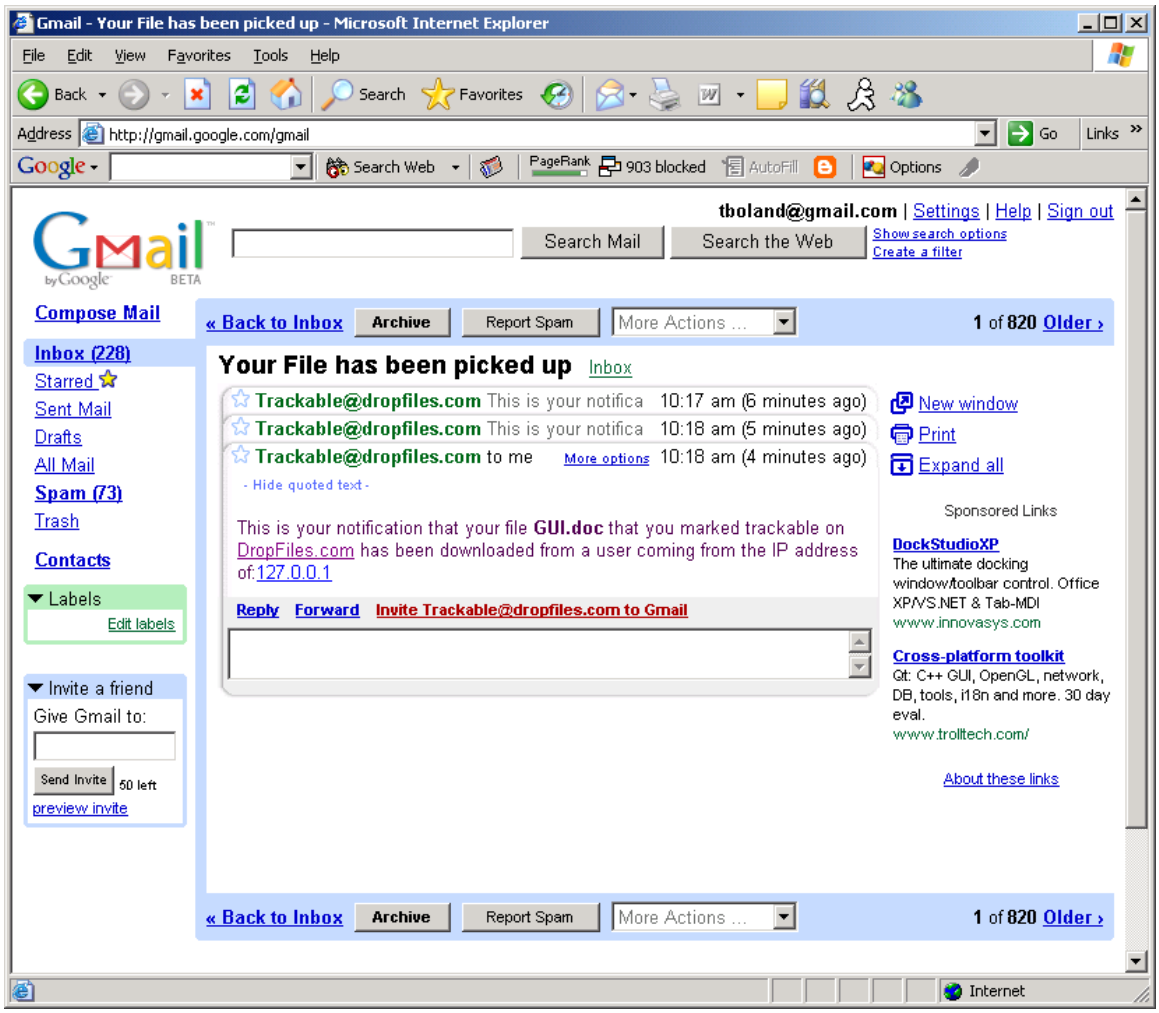


Figure 11. Tracking Email Sent to Member

4.4 Management Interface

The management interface was done using a .NET datagrid with buttons and columns. Memebers are able to see what files they have uploaded on the system, how big the files are in terms of Megabytes, their total space usage, and the ability to delete files. The delete method does not actually delete the file from the hard drive, but instead sets a flag in a SQL table. Later a cron job like service will be created to delete files that have a flag marked. Below is the management interface.

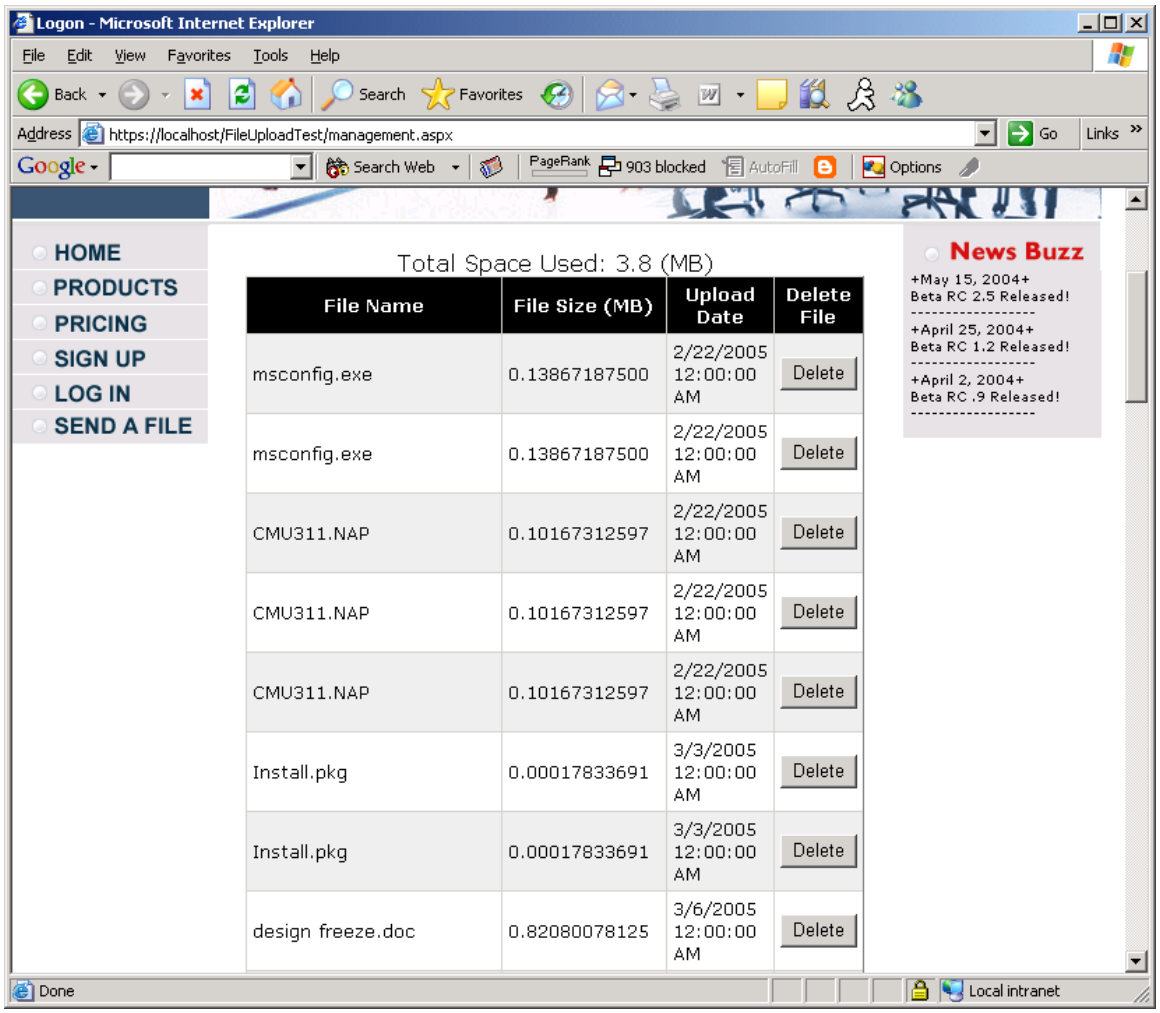


Figure 12. Member Management Interface

4.5 Web Service API

The web service API was created to allow other software and web applications the ability to tap into the main functionality of the system. Right now the web service API takes two arguments, file name and email address along with the file stream. I used Direct Internet Message Encapsulation (DIME) to attach the file to stream it along with the SOAP header. I created a simple client web application that uses the web service API to demonstrate its functionality. Figure 13 a screen shot of that example.

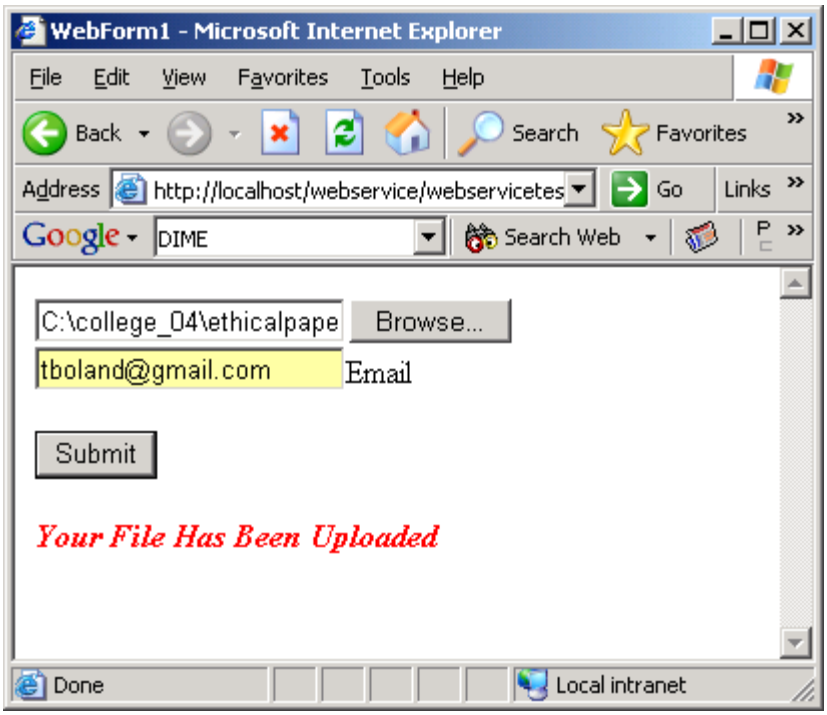


Figure 13. Client using Web Service API

5.0 Testing Procedure

Two software packages are being used to run load, functional, and regression testing on my web application. To do load testing, I am using Microsoft's Web Application Stress software. It allows you to record a script, which is a user interaction on a web application and then run those same interactions acting as hundreds to thousands of users, showing response time by the server. It is very important that my application be able to support hundreds of concurrent users at the same time. For functional and regression testing I am using Seapine Software's QA Wizard 3.0. QA Wizard allows me to create automated scripts to run against my web application, putting in different parameters to test functional response and for acceptable error handling of the application. I first record a script that demonstrates all intended usage of the web application. Then in QA Wizard I am able to arbitrarily change input and any other interaction with the recorded script, and then play back that script with those changes

testing functionality. All scripts produce a XML based report with failure/succeed documented. Figure 14 shows QA Wizard doing testing on dropfiles.com

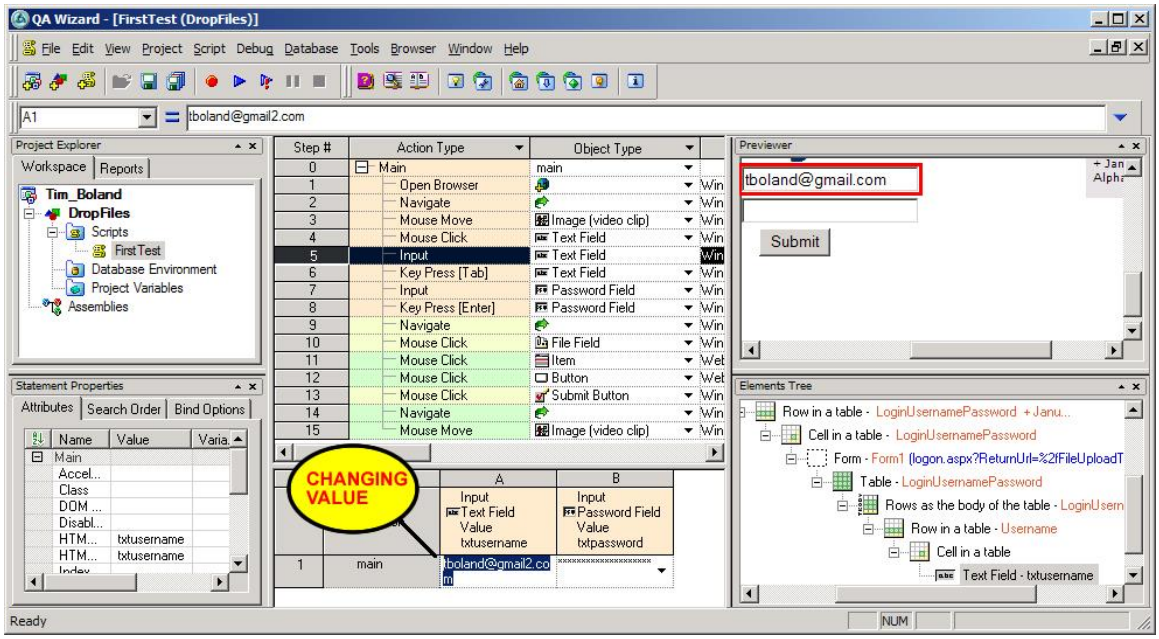
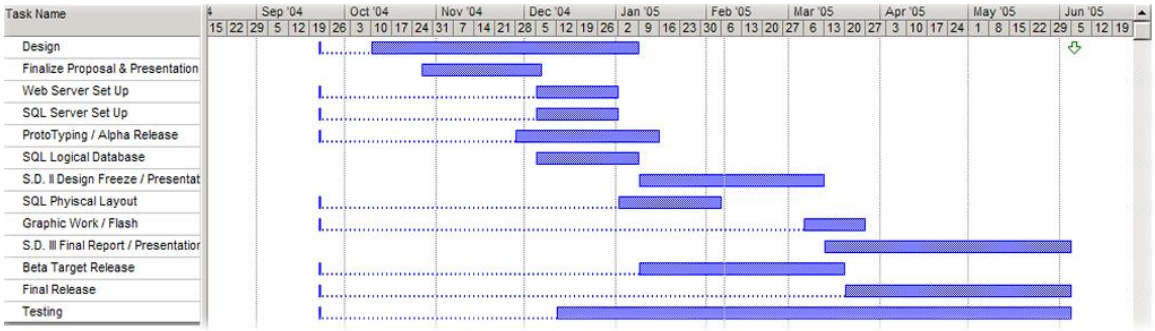


Figure 14. Testing logon authentication

6.0 Development Effort

6.1 Timeline

The timeline scheduled was strictly followed during this project. I used Microsoft Project 2003 to help coordinate deadlines and overall progress of the project. Below is the timeline used in Microsoft Project.



6.2 Budget

Included in the budget is the hardware and software that I used developing DropFiles.com.

Item	Explanation	Cost
Microsoft Visual Studio.NET 2003 Professional	University License Agreement	$899.00
Microsoft SQL 2000 Enterprise Edition	University License Agreement	$3,899.00
Flash MX 2004 Pro Professional	Need	$300.00
GIMP	GPL License	Free
Pentium III – 750 MHz, 512 RAM, 40 Gigabyte HD	Already own	$395.00
Microsoft Server 2003 Web Edition	University License Agreement	$400.68
		Total: $5,893.68

7.0 Conclusion and Recommendations

7.1 Conclusion

After spending over two years working in marketing and observing the need for companies to be able to exchange large files simply and securely, DropFiles.com accomplishes this goal with additional features that exceeded the original idea. The object oriented design helped tremendously throughout the project by being able to reuse class structures and maintain a consistent code base. Coming from a Java and PHP world, I found some of the ADO.NET very useful such as datagrids.

7.2 Recommendations

During this project I learned a great deal about sessions, database design, and project management. The web application is generic enough where the system could be easily ported to a LAMP solution. For the application to go into production mode I still need to integrate an instant payment system such as Paypal. This should only take 5 hours to setup once the system is on a static IP address. A fault tolerant server setup still needs to be put into place. My recommendation is to set up a three hard disk setup using striping with parity to ensure fault tolerance.

References

1) University Virginia. “Clear Text Passwords and Windows Security.” http://www.itc.virginia.edu/homedir/security/plainpass.html#what. October 3, 2001.

2) University of California Technology. “Email Inbox Quota, Message Size Limits and Attachment Formats.” http://www.its.caltech.edu/e-mail/quota.html November 11, 2004.

3) Security Management. “Hashing Out Encryption Solutions” http://www.securitymanagement.com/library/001050.html. June 2001.