Professor Guillermo F Salazar **June 18, 2008**
Civil Engineering Department
WPI
Worcester, MA 01609

Dear Professor Salazar:

Attached is one copy of the Interactive Qualifying report: **EDUwebMaker: A Prototype Website Generator for Schools**, Project Number GFS 0002.

Sincerely ,

David A Croswell
Mark H Hassett
Matthew F White

Distribution: Library: 1 copy
Professor Guillermo F Salazar: 1 copy

Project Number: GFS 0002

EDUwebMaker: A Prototype Website Generator for Schools

An Interactive Qualifying Project Report

submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Bachelor of Science

by

David Croswell _____

Mark Hassett _____

Matthew White _____

Date: May 20, 1988

Approved:

_____

Professor Guillermo Salazar, Advisor

ABSTRACT

The resources required for the initial development of a website for a private school with limited financial resources prevents many of these schools from reaping the benefits of this technology. This project researches schools' needs for websites and develops prototype software of a low-cost website generation tool that may be used by the faculty of these schools to generate their web sites. The functionality of this prototype is tested by potential users in terms of quality and simplicity with promising results.

AUTHORSHIP PAGE

David Croswell

- Section 4 - Results
- Section 5 – Analysis of Results
- Section 6 – Conclusions and Recommendations
- Results of Survey

Mark Hassett

- Section 1 - Introduction
- Section 2 – Background Information
- Survey of Schools

Matthew White

- Section 3 - Methodology
- Section 4 - Results
- EDUweb Maker software
- EDUwebMaker data files

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

EXECUTIVE SUMMARY

The internet has become an important resource for gathering and distributing information. It can be used as a way to communicate with many people in a very easy and efficient manner, however, if one lacks the knowledge of how to use the technology it can be a difficult and frustrating experience. The internet, or the World Wide Web, in particular, is a very useful technology for schools. Many universities, such as WPI, have a wide variety of administrative and academic information available through the Web. This allows students to access information directly related to courses they are taking, activities during the week, or sports schedules, very easily at any time. These schools have invested thousands of dollars in setting up and maintaining these resources, which could drive up costs for students or require funding from the government. Similarly at the high school level and below, the web is still a highly valuable resource, and many schools invest in the technology necessary to make use of the web for academic purposes. Public schools can use public funds for developing a web site to keep students and parents informed of school activities, athletics, etc. as well as providing academic resources such as course information, class schedules, and anything else that students may want to access both at school and at home. Some private schools lack such great resources, however, and so they don't always have the necessary funds for such technology, which can limit their competitiveness in the academic arena. Religious schools in particular, which might receive little or no public funds, often run on tight budgets that make developing and maintaining an academic and administrative website for their school something that can't be done very effectively. The concept implemented in this project as an automated website generator explores the possibility of such schools being able to easily create feature rich web sites which provide quality communication between the school, parents, and students. The software is designed and tested to be easily used by any teacher or staff person who may have limited knowledge of internet technology.

1. INTRODUCTION

The World Wide Web is one facet of the vast information network we call the internet. Sometimes called the information superhighway, the internet is an evolving technology, constantly changing, molded to deliver information faster and more efficiently. While there are many methods of accessing information via the internet, the web is one of the most commonly used today. This is likely due to the fact that a graphical web browser interface is very user-friendly, with a point-and-click operation that does not require knowing all sorts of commands which other internet portals such as telnet and gopher use. As a result of the popularity of the web, there has been a plethora of information created for the web in the web standard Hypertext Markup Language (HTML), as well as electronic copies of pre-internet information, stored on servers world wide and made available to anyone with the tools necessary to access it.

One of the advantages of the web as an information resource is its global nature. Anyone with web access can get information from any web server in the world at any time, and in many cases free of charge. It is like having an enormous public library on one's desk that is open all day, every day of the week. In fact, some public libraries, and other libraries as well, have web sites allowing access to online resources both during and outside of their operating hours. But the information on the web reaches far beyond what a library can contain, as any computer connected to the internet can act as a web server, delivering information about anything to anyone who requests it.

This freedom of information access and distribution raises many issues, however, not the least of which is the issue of information security. A reputable web site requires some sort of security to protect its information from alteration by anyone not authorized to make changes to the content on the site. There are many threats to computer systems open to the public, such as any computer connected to the internet. Malicious users both physically present at the computer or using another computer anywhere else in the world connected to the internet can and do try to cause harm to computer systems, be it distributing a virus, defacing a website, or destroying information. Good security can prevent most of these intrusions, but the hardware and software required for such protection is usually prohibitively expensive for an individual to invest in. However, an individual usually does not have the need for their own web server. This is why most personal web sites are on shared servers where a company owns the hardware and

software for serving the web sites, and rents space to individuals or other companies. This makes efficient use of their systems, as well as makes it affordable for many people to have their own web presence.

There are drawbacks to these shared servers, however. They are generally restrictive in regards to how much information and what types of information can be stored. They often require the user to have knowledge of HTML or at least have software capable of generating HTML pages, although many do offer very simple web site creation tools. The main problem with such tools is the limitations they place on what a user can do and what technologies they can use in their web sites. Because of all the limitations of shared servers, many organizations choose to operate their own servers for serving their web sites, which allows much flexibility and creativity, but at the cost of purchasing and maintaining equipment, as well as employing software developers to create and maintain the web sites.

The commercial advantages to having product and service information available on the web are obvious: it is one more advertising venue in which the company can invest. Most, if not all, large commercial organizations own and operate their own web servers in order to guarantee the security of their information, as well as to have the flexibility of deciding what equipment is necessary and upgrading when necessary. There are many other advantages as well, all of which are aimed at increasing the company's profit. Many non-profit organizations also invest in their own servers for similar reasons. Charitable organizations may use their web presence to advertise to increase awareness for their cause. Academic organizations may recruit students and inform alumni of news and events. Organizations of all types and sizes have staked a claim to space on the web to allow them to share information of all sorts with the rest of the world across the internet.

The web is a very useful technology for schools. It is an immense resource for gathering information, and for distributing information. Many universities, such as WPI, have a wide variety of academic information available through the Web as well as other less common internet protocols such as Telnet. This allows students to access information directly related to courses they are taking, or plan to take, very easily at any time. These schools have invested thousands or millions of dollars in setting up and maintaining these resources, but this can be offset by increasing the cost of tuition, or, for public universities, obtaining more public funds. Similarly at the high school level and below, the

web is still a highly valuable resource, and many schools invest in the technology necessary to make use of the web for academic purposes. Public schools can use public funds for developing a web site to keep students and parents informed of school activities, athletics, etc. as well as providing academic resources such as course information, class schedules, and anything else that students may want to access both at school and at home. The schools may even be able to share hardware, software, and development resources with the local government to defray these costs. Private schools, however, don't always have the necessary funds for such technology, and this often limits their competitiveness in the academic arena. Religious schools in particular, which might receive little or no public funds, often run on tight budgets that make developing and maintaining an academic website for their school difficult to justify. Renting space on a shared server is one inexpensive option, but does not address the cost of developing and maintaining the website, which normally requires at least one person working full time. An additional employee, especially with the skills required for developing and maintaining a website, is a major expense for a limited budget. This usually means that the school either has a very limited, poorly maintained website, or does without completely. Of course a school does not need to have a website, but if the cost could be brought within reach of the budget limitations, the advantages are tremendous.

There are several ways in which the cost of a website can be reduced. One is to reduce the size and/or complexity of the site, and therefore reduce the time needed to develop and maintain the site. This results in both lower development costs and lower maintenance costs, but at the expense of greatly diminishing the utility of the website. Another alternative would be to utilize one of the many "user-friendly" web development tools on the market today, so that an outside consultant or additional faculty member would not have to be brought in to do the development and maintenance. These tools include programs like Microsoft FrontPage, Macromedia Dreamweaver, and other similar products. While these programs allow a high degree of flexibility and advanced features to create a highly functional website, they still require a substantial knowledge of web technology to create such a website. These products are also quite expensive for just a single copy (approximately $149.99 for FrontPage, $319.99 for Dreamweaver) [1], so with a limited budget a school could probably only afford to buy one copy of the software, if it could even afford that, and so all website development and maintenance would be limited to a single computer on which the software was installed. These two options place significant limits on the utility of the website or the cost and ease of use of the website. Ideally,

there would be an alternative that would be inexpensive and yet still allow the creation of a robust website that will enable a school to take full advantage of web technology. This solution would also allow maintenance to be performed not just at one computer with the special software installed, but from any computer with access to the website. This is the type of solution proposed by this project.

In order to demonstrate the demand for a solution such as described in this report, we created a survey to get feedback from schools. In this survey we asked whether the school had a website, what features they would like to have on a website, and whether they would use a product which allowed them to implement such features. We distributed this survey to 49 Christian schools throughout the US based on a directory of Christian schools found at http://www.acsi.org [2]. The responses we received are detailed in Appendix B, and the graphs below in Figures 1, 2, and 3 summarize that data.

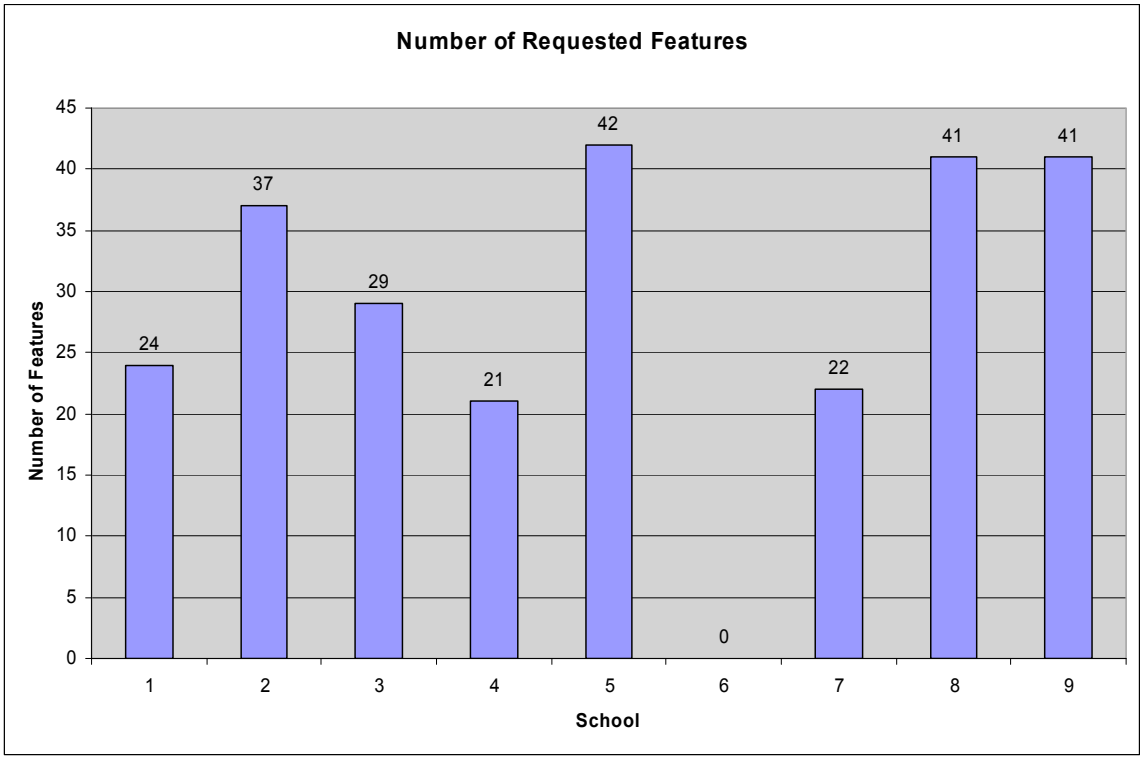


Figure 1 – Number of Requested Features

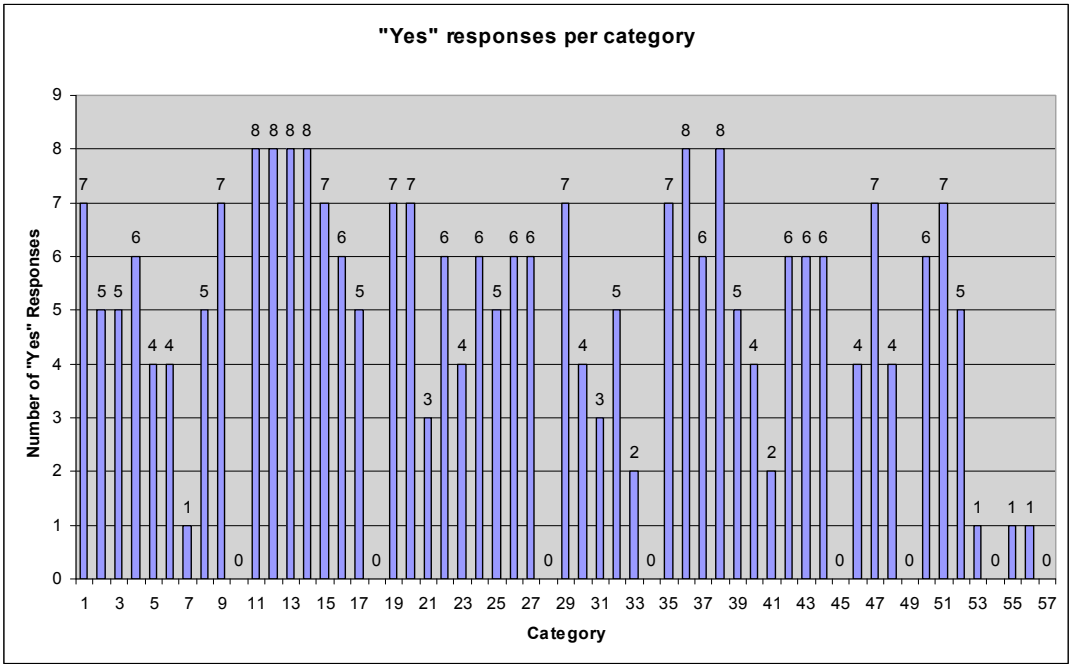


Figure 2 – Yes Responses per Category

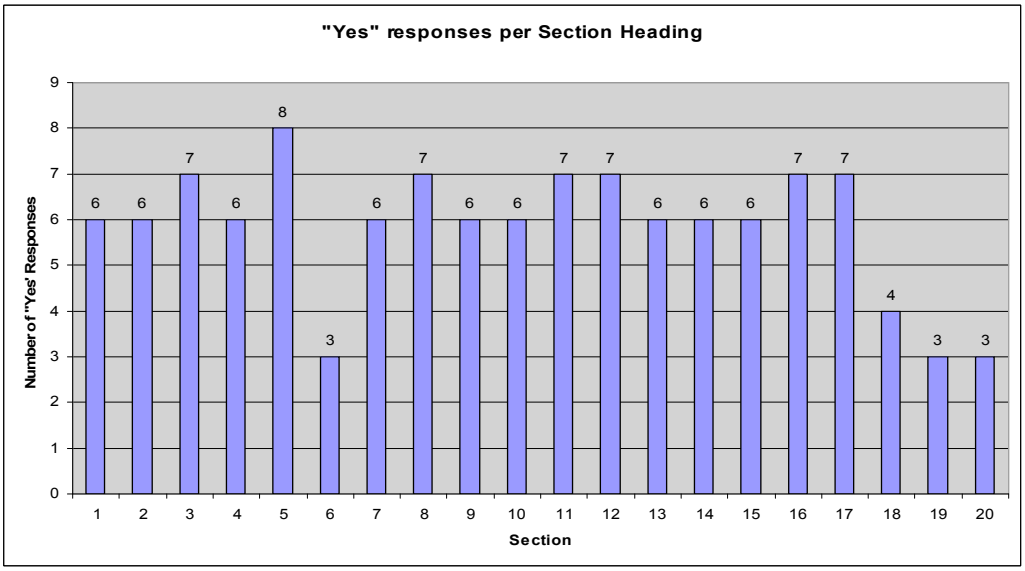


Figure 3 – Yes Responses per Section Heading

Of the schools which responded, 60% already have websites, but 67% of these were interested in the ability to add features and functionality to their website, as our solution would allow them to do. A few expressed a desire for specific features which their current solution could not accommodate, or perhaps could only accommodate at a prohibitively expensive cost. The other 40% of schools did not have websites and were interested in developing one, and they also liked the ideas we proposed. From the enthusiastic responses we concluded that there was a strong desire for a highly flexible yet easy-to-use and inexpensive website development and maintenance solution.

2. BACKGROUND INFORMATION

2.1 Web Sites Today

Web sites today have evolved from the early text-only pages into complex documents using graphics and scripting to create an aesthetically pleasing site that can deliver the necessary information to the user as quickly as possible. A good website is easy to navigate, providing an obvious path to the information a user wants. Often web sites make use of databases to index the information available on the site so that a user can search for their desired information quickly and easily. Increasingly sites use these databases not only for a search feature, but also for storing the information in a highly organized fashion, which allows the web server to make use of technologies such as Active Server Pages (ASP), Java Server Pages (JSP), CGI scripts, or other database retrieval technologies to dynamically create web pages with the information stored in the database. This allows the display of information the user requests, as well as other potentially related information that may be of use to the user. Because the emerging technologies based around the Extensible Markup Language (XML) are found in both databases and web browsers, the merging of these two is becoming much more common. XML is related to HTML in that they are both markup languages, but XML is much more flexible and can be easily adapted to many applications.

The use of databases to store data for a web site makes sense, as a database is designed to store data, and a web site is designed to deliver information. It can also be much more user-friendly on the developer side to only have to update the database to change information on the web site. This avoids the need to go through many lines of HTML code to find where the information needs to be changed, and if the information is on several different pages, repeating the process for all pages can be very tedious. One simple change to the database updates all the pages on which that information is used. This allows those who are unfamiliar with HTML and other web technologies to make the necessary changes so that web developers can have more time to incorporate new features into the web site.

Security becomes an even greater issue when dealing with databases, however, because not only do the web pages need to be secured from unauthorized modification, but the database also has to be secured. At the same time users must be able to

access the information freely, and those authorized to make changes must be able to make any changes they need to make. These issues usually are much more significant for larger, high profile web sites that are more likely targets for hackers, but still must be kept in mind by all web site operators.

2.2 Christian Schools

Today, more than ever, there is a need for an organization to have a strong web presence. This need is felt, even by schools, as the number of families with computers increases and the World Wide Web becomes the most prominent source of information for these families. Schools, private Christian schools in particular, often don't have the resources or the means by which to create a "web-presence" and are therefore falling behind in this technology-centered society. These schools, while they often cannot afford the expenses to create a feature rich website, would benefit immensely from the use of one. A feature-rich web site might provide such information as a calendar of events, class assignments, bulletin boards, school information, and notices of school cancellations or other important news. The cost to create such a website that allows up-to-date information can often be overwhelming to these schools. Other costs which can also inhibit the use of a feature-rich up-to-date website are the maintenance and technical support fees which can be very high as well. Therefore, it would be beneficial to these schools to have a program which would enable them to generate a complete web site that has all the advantages of a professionally made site without requiring any programming experience; one that would allow the web site to be easily maintained.

These schools would particularly benefit from the type of program proposed by this project. As explained earlier, the use of a database as the information storage mechanism removes the need for a web developer for updating the content of the web site. The program would also offer a flexible structure with many options, allowing the school to choose appropriate themes for their school. By handling the navigation menus dynamically through the database, the web site is guaranteed to have excellent navigation and allow quick and easy access to the information any user may want to see. Additionally, the ability to add and remove features gives a school the option of future expansion, or the option of removing a feature should they find they no longer have a need for it. This ease of use and flexibility both speed up and simplify the creation and management process, as well as allowing any of the school faculty and administrators to make

changes to the portions of the web site to which they have access. By distributing the management load, it becomes simple to set up and maintain a good, robust web site.

2.3 School Websites

There have been several governmental initiatives at the federal, state and local levels throughout the US for technology funds for education. These represent an enormous government investment in using technology to improve education. Many such funds are outlined at http://www.ed.gov/Technology/tec-guid.html [3]. Below is just one except demonstrating the size of the technology investment:

> ***Technology Literacy Challenge Fund***
>
> > A five-year, $2 billion fund to provide formula grants to state education agencies to support grassroots efforts at the state and local level to meet the four national technology goals for schools: modern computers, high quality educational software, trained teachers, and affordable connections to the Internet. After the Technology Literacy Challenge Fund was launched in FY97 and funded at $200 million, it more than doubled with $425 million appropriated in FY98. These funds are awarded to all 50 states and outlying areas according to the ESEA Title I formula. The challenge fund is designed to encourage states, local communities, the private sector, schools and individuals to work together to integrate technology into teaching and learning.

Of course, most of these funds go into equipment purchases and not into the development and maintenance of websites for schools, but the substantial amounts available dedicated exclusively for technology purposes allows for some funds to go to a web development initiative. In addition to these federal funds, many state and local governments have similar programs for supplemental funds for technology investments. Because of the disparity in tax revenue and cost of technology deployment, poorer school districts tend to have less opportunity for technology development than more wealthy districts, but there is at least some funding dedicated for technology programs.

For private schools, there is a similar disparity between poor and wealthy, but there is no governmental equalizer to funnel funds to the schools with smaller budgets.

Private schools, particularly religiously affiliated private schools, usually get some income from tuition payments and some from charitable donations from alumni or community members. Thus, if these schools want to invest in technology, the funds need to come from one of these sources. For example, some schools have numerous distinguished and wealthy alumni who have contributed very substantial funds for a wide variety of purposes, some of which involve technology investment. As a result these schools are able to afford faculty and staff members who work full-time on development and maintenance of an extensive website. On the other hand, a less fortunate school may get very little in donations, forcing either little or no technology investment, or an increase in tuition costs, which their students may not be able to afford. Certainly a small school with only a few faculty members could hardly afford the substantial investment in an additional staff member to handle the tasks of website development and maintenance. Someone with such high-tech training would likely command a significantly higher salary than the average teacher's salary. Consequently, such schools tend to lag behind other schools, both public and private, in utilizing new technology. To enable them to make effective use of web technology, a high-quality, low-cost solution is necessary, and can be achieved by developing a solution which targets that specific need.

3. METHODOLOGY

A straightforward methodology was used to realize this project. First, we identified the problem. We did this by composing a series of survey questions that we then sent to 49 private Christian schools throughout the United States in October 2000. Ten schools sent completed surveys back to us. The survey mostly asked what features a school would want on their website, if they currently have a website, and if they would like to be able to easily and dynamically update the content on their site. We also researched the current state of a dozen schools by visiting their websites. We came to the conclusion that many schools have very basic and content-poor sites, or no sites at all. A reason for this was that the schools couldn't afford the large amounts of money required to pay a company to build a website for them, which, as of May 2000, would cost anywhere from $113,500 to $608,000 [4].

Next, we devised a solution to the problem. The solution was to build a software program, EDUwebMaker, that would generate the complex web sites needed by today's school, yet require only very limited technical skill, and also be two orders of magnitude cheaper than having a professional build the web site. In fact, the software would do all the work, so the price could be set to whatever profit was required by the company selling EDUwebMaker. Even if the software cost a couple thousand dollars, the resulting website would be two orders of magnitude cheaper than having a company design it for you. Not only will this solution be cheaper, but it will also have the ability to offer the web pages that many schools mentioned they would like on the survey. We determined we would make a prototype, sort of a proof of concept, of our solution. This would allow us to test the feasibility of the idea. This would allow us to focus on the idea of the solution, and not on many of the details that come with making a finished product.

The specifications, particularly the software specifications, were then developed for the program we had determined would solve the problem. These specified exactly what the software needed to do, so that at the end, we could determine whether we had met our goals or not. From the specifications, the software was actually built. Additionally, the support files needed by EDUwebMaker were also create to specification.

At this point, testing began. First there was function-level testing, where individual functions in the software were tested to insure they performed as expected. Secondly,

functionality testing was performed to see if the software met specifications. Many software bugs were found in these first two stages that required significant time to correct.

Finally, user testing was performed. The software prototype was used by two school employees, who tested it and commented on the concept of the software program. Basically, they were testing the usefulness of the software. The quality of the website generated was then compared to the quality of a website which would be created by a professional website developer, and the prices were also compared.

In the end, the work was documented. This is especially important, since we only created a prototype. This software could be marketed, judging from our feedback, but that would require more development, something that would have to be done at a later date. From our documentation, others should be able to continue where we stopped.

4. EDUwebMaker

4.1 Overview

The website generator created by the authors, known as EDUweb Maker, is the technical proof of concept that shows how the idea of a dynamic website generation program could be constructed. The goal of EDUwebMaker was to create a program that would contain all the features of a full-fledged school web site generator; however, the program would not include all the options that a full program would contain, and it also wouldn't be tested to the same level of quality as a final product. This is what makes it a proof of concept. It proves that the concept would work; creating a final solution from this proof of concept would be straightforward, albeit time-consuming.

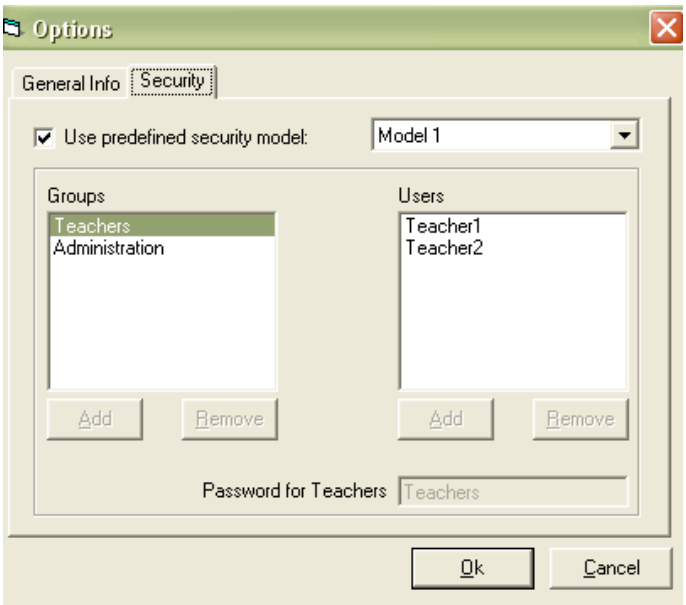
EDUwebMaker is a software program that the user installs onto their computer. When the program is executed, it allows the user to create a completely dynamic school web site based on HyperText Markup Language (HTML), Active Server Pages (ASP),

Figure 4 – Screen shot of EDUwebMaker program

and Open DataBase Connectivity (ODBC). It outputs the web site as a series of files in a given directory. These files can then be uploaded to any web server running Microsoft Internet Information Server© (IIS) to allow the web site to be accessed from anyone with access to the World Wide Web (WWW). EDUwebMaker will be designed to work on any Windows 95 or later based computer. A screen shot of the EDUwebMaker program can be seen in Figure 1.

An overview of how a session of using EDUwebMaker might go is as follows. After installing EDUwebMaker using the setup program included on the CD, the user executes the installed program. The user selects to create a new project and then sets the project folder. This is the location where the generated files (that need to be uploaded to the server) will be placed. Next the user enters some basic information about the project (for example, the name of the school), and then describes the desired security model (or picks from some predefined ones). The security model defines the user names and passwords for each person who will have access rights to modify the web site. This security model can, of course, be updated later. An indefinite number of groups of users can be placed in the security model. A screen shot of the Properties window that allows the user to define a security model is shown in Figure 2. In the figure, there are two groups, Teachers and Administration. As an example, Teachers would hold all the

Figure 5 – Screen shot of Properties window



teachers as individual user. Administration would hold users who would have access to

Figure 6 – Screen shot of Options window

higher-level sections of the website.

Next, the user selects the options for the web site. This includes the layout, navigation, style, features and effects. The layout describes the general layout of all the web pages (where the navigation bar is located, etc.). Navigation lets the user select which navigation bar style they would like. Style selects the look and feel of the web site, including the colors, backgrounds, and icons. Features selects what web pages will be included in the site. Finally, effects selects any special effects you would like on your site (and is limited only by the creativity of the programmer). For example, it could display a special message to the user if it recognizes the person viewing the website.

Next, the user selects to generate the web site. The web site is generated, a default database is created, and the files are all placed in the given directory and are ready to upload to the server. However, before uploading them the user can test the web site. This displays the web site in the project window exactly as the user will see it. Not only does this allow the user to check that the web site looks ok, but it also allows them to go through the administration pages to enter the data that should be initially on the web site. The administration pages are available to anyone that can get to the web site. However,

a user name and password are required, and are stored in the database. The administration pages allow the data on each web page in the web site to be updated. The updated data goes in the database. At this point, the user uploads the files to the web server, and the web site is up and running!

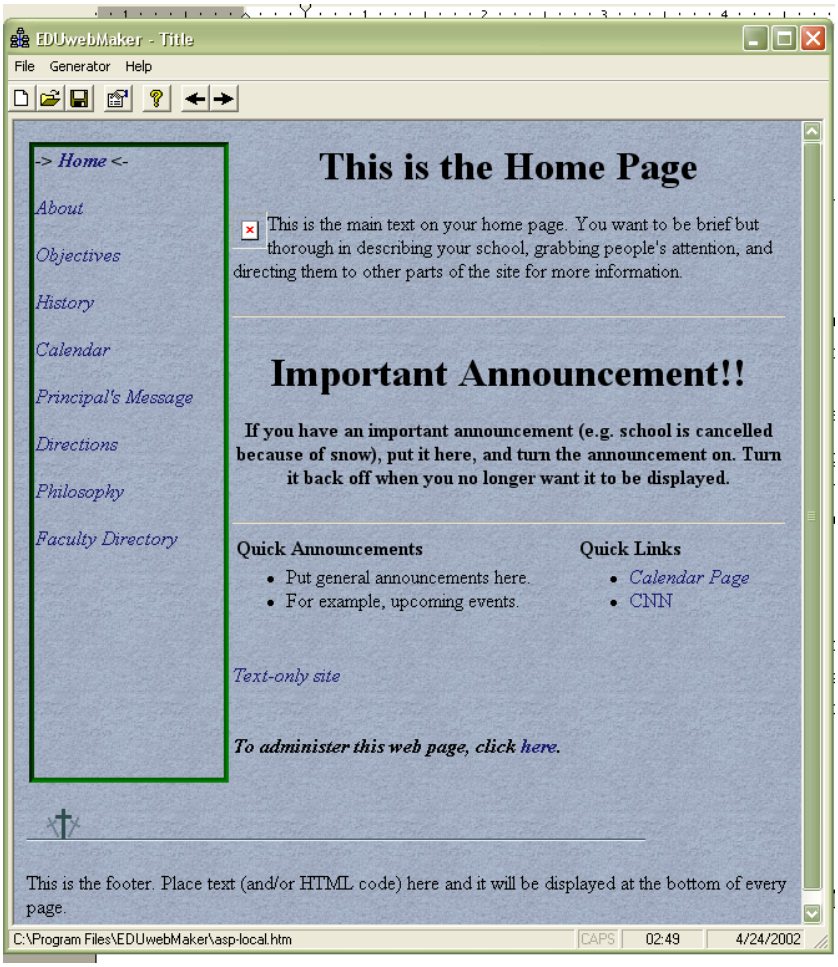


Figure 7 – Sample Home Page

In Figure 4, a sample web page is shown, as displayed by EDUwebMaker after it has generated the website. The fact that the navigation bar is on the left and is surrounded by green is determined by the "layout" choice. The style of the navigation bar, like the fact that it is vertical and has arrows pointing at the current page, is determined by the "navbar" choice. The background texture and the styles and colors of the fonts are determined by the " style" choice. The actual layout of the current page, and the information on it, is determined by the "feature" choice. Notice also that at the bottom of

the page is a link to the administration section of website, where administrators can update the content on the website.

4.2 Features

EDUwebMaker contains many important features that will be described below. First, it is very powerful. It can generate a dynamic website that is served its dynamic information through a database. The data on the site can be customized depending on the time or who is looking at the site, for example. By using a database, data to be displayed on the web site can easily be stored and retrieved. EDUwebMaker is also flexible, both from a user's point of view and a programmer's. In this context, the user is defined as the person at the school who generates the web page, and the programmer is the person who designs the pages that the user can select (typically, the programmer would work for the company that sells EDUwebMaker). The user can change everything from the style to the layout to the contents to the security model used. The programmer, on the other hand, can easily add "plug-in" modules that expand the options the user gets. Adding options becomes very straight forward. Other benefit of EDUwebMaker is it's ease of use. Even though it creates web sites that would normally take a professional web developer half a dozen months to develop, it should be able to be used by someone with very limited technical knowledge. Finally, the program creates secure web sites; that is, websites that cannot easily be modified by the wrong people. This attention to security allows the web site to be updated from any web browser by different people, where each person has their own user name and password that give them access to a certain part of the site.

4.2.1. Power

EDUwebMaker supports ASP, a way to make dynamic web pages (i.e. web pages with content that changes depending on the current time or the current user). ASP can read data from a database and then display that data on the screen. EDUwebMaker can do just about anything that ASP can do. It uses ASP to get its information from a Microsoft JET database, which is the database technology used in Microsoft Access. The database is created when the web site is generated, and can then be updated by

the administrators as they make changes to the web site. A user browsing the web site sees the data that is currently in the database, thorugh the web page. This allows the information on the web site to be dynamic, not static. For example, one could have a calendar of events. A table in the database would be created, when the web site is generated, for the calendar of events. An administrator could then access the online administration pages and tell them to add a new event. This new event would be placed in the database. When a user then goes to view the calendar web page, the user gets the data currently in the database, and so sees the updated calendar of events.

A second useful feature of ASP that EDUwebMaker uses is the ability to present information that meet certain conditions at certain times. For example, a calendar web page might only display events for the current month, or an announcement on an announcement page might only be displayed on the web site for a week. Being able to schedule when information will be displayed makes it much easier for those people maintaining the web site. They can schedule when information will be available, and then forget about it.

A final useful feature of ASP utilized by EDUwebMaker is the ability to display different information to different users. This is primarily useful for the security aspect, mentioned below, where each administrator only has access to a certain part of the site depending on their user name and password. This feature could also be used by users of the web site, though. For example, students could be given user names and passwords that would allow them to see their grades.

4.2.2 Flexibility

EDUwebMaker is very flexible. All the options (layout, navigation, style, features and effects) are all independent of each other and can be mixed and matched in any way. Any amount of detail can be easily added to each web page template by the programmer. This will give the user as much control as possible over each page.

Although this is the most obvious benefit to the user, the more important benefit may be the flexibility to the programmer. All the options that the user can select are extensible by the programmer. For example, all that's needed for the programmer to add a new layout is to create a new ASP file and then reference the new file in the database. The programmer needs not have access to the original source code for EDUwebMaker.

A second advantage to this is the ease of creating add-ons to the software. Someone, for example, could make their own set of styles, layouts, features, etc. available as add-ons to the original software. These could then be sold as upgrades to the user.

4.2.3 Ease of Use

EDUwebMaker is easy to use. The user need not know how the technologies behind the program work. They don't even need to know HTML. It is possible for someone with very little technical knowledge to design the layout of the website and add the data to be displayed on it. The user must only know what they want the layout of the web site to be, and the data they want displayed on it.

All the technical details are hidden behind the interface. The user is presented with the web-site design level, not the coding level. They choose how the web site looks from a list of options, which pages they want on the site from a list of options, and then proceed to enter the data into the database via web-based forms. Someone familiar with the WWW should be able to design a web site using EDUwebMaker.

4.2.4 Security

One important feature is security. It is imperative that unwanted persons cannot modify the generated web site. This is a very real concern since the administration of the web site is offered **through** the web site. The administrator logs on to the administration section of the web site using a user name and password. Administrators are then presented with a list of options they can modify. What they have access to is determined by their login name and password. Without this user name and password, someone can't make any modifications to the web site.

The security model is set up as follows. There is a main, built-in administration account, called *admin*. This account has a default password of *admin*, something that should be changed immediately. The *admin* account has complete access to everything: all the web pages, and complete access to change the security model, including changing passwords, and adding and removing groups and users. The *admin* account is the only default account. That account can be used to add more accounts later, or they can be specified in the security settings when a project is first created.

Groups and users can be created. For example, a *Teachers* group could be created, and then each teacher could get a user account within the *Teachers* group. A group account has permissions to certain web pages, which are inherited by the user accounts within a group account. A group account also has a password. Someone can log in using the group name and password, and then make changes to the membership of the group.

Users can be added to groups, in which case they inherit the permissions of the group they are in; they can be given additional permissions, as well. Users can be created outside of groups, as well. And example of a security model is as follows:

admin	- default account; permissions to all pages and all users
Teachers	- group account containing all teachers; permissions to the Teacher's Overview page and all users in the Teachers' group
--- Fred	- user account for one of the teachers; permissions to the Teacher's Overview page and Fred's personal page
--- Mary	- user account for one of the teachers; permissions to the Teacher's Overview page and Mary's personal page
Secretary	- user account (not in a group) for the school secretary; permissions to all the general school information pages

4.3 Working Prototype

As stated before, the number of options in EDUwebMaker is very small, though all features are demonostrated; only enough options are provided to prove that it works. This is what is currently there. There is a simple installation program that installs EDUwebMaker and components required to make it run on the given operating system (it supports all Windows-based machines). Also included is a readme file for basic installation and execution instructions.

The program itself comes with some basic options to show the potential of the EDUwebMaker program. The main part of the screen is an HTML browser. This is useful for previewing the web site you are creating, but it also allows context-sensitive help to be given in the form of HTML pages while you are creating the site. The program pro-

vides help to the user for each step in the process of creating the web site. Additionally, some additional help can be accessed via the “Help” menu.

The program lets you create, open, and modify “projects”. A project is a set of the options that a user specified for a website, along with all the generated files for it. Once you create a new project, you may select the options for the web site. Here are the basic options that show off the power of EDUwebMaker. In a full product, there would be many more options, and much more detail for each one. Under Layout, you select the basic layout of the web pages on the web site. There are only two available in this proof of concept software: navigation bar on the left or on the top. Under Navigation bar style, you may select various styles of navigation bars for your site. There are currently four simple ones. Under style, you set the look and feel (colors, graphics, etc.) of the web site. Four simple schemes are provided to show what could be done.

There are ten sample web pages under Features. Each of these becomes one web page in the web site. “home”, “home2”, and “calendar” are complex pages with highly customizable content. The other seven pages are simple, just displaying information about the school. “home” and “home2” are two possible home pages, with home2 being the more complex of the two. The user can select which page is the startup (home) page by selecting it from the drop-down list at the bottom of the window. The calendar page displays a three-month calendar with events listed on the day that they occur. Note that the calendar is time-sensitive, displaying the current month and the next two months. “History” displays a page that includes the history of the school. “Objectives” displays a list of goals that the school has. “Principalmessage” displays a message from the principal of the school. “About” lists information about the school. “Directions” provides directions to get to the school. “Facultydir” lists the names and e-mail addresses of the faculty. “Philosophy” displays the philosophy of the school.

Finally, the user may select any effects you want on your web site. The current choices are “none” or “Status Bar”. The status bar option scrolls a welcome message in the form “Welcome to <school name>!” at the bottom of every web page. Effects are written in JavaScript and can be as complex and creative as the programmer of the effects wants them to be.

Also included are basic administration pages for each web page you can select. They let the user insert and modify all the information displayed on that page. The user interface isn’t very pretty for the administration pages, but they get the job done. In a

final implementation of this program, the administration pages could be made much more aesthetic and user-friendly. An example of an administration page is shown in Figure 5.

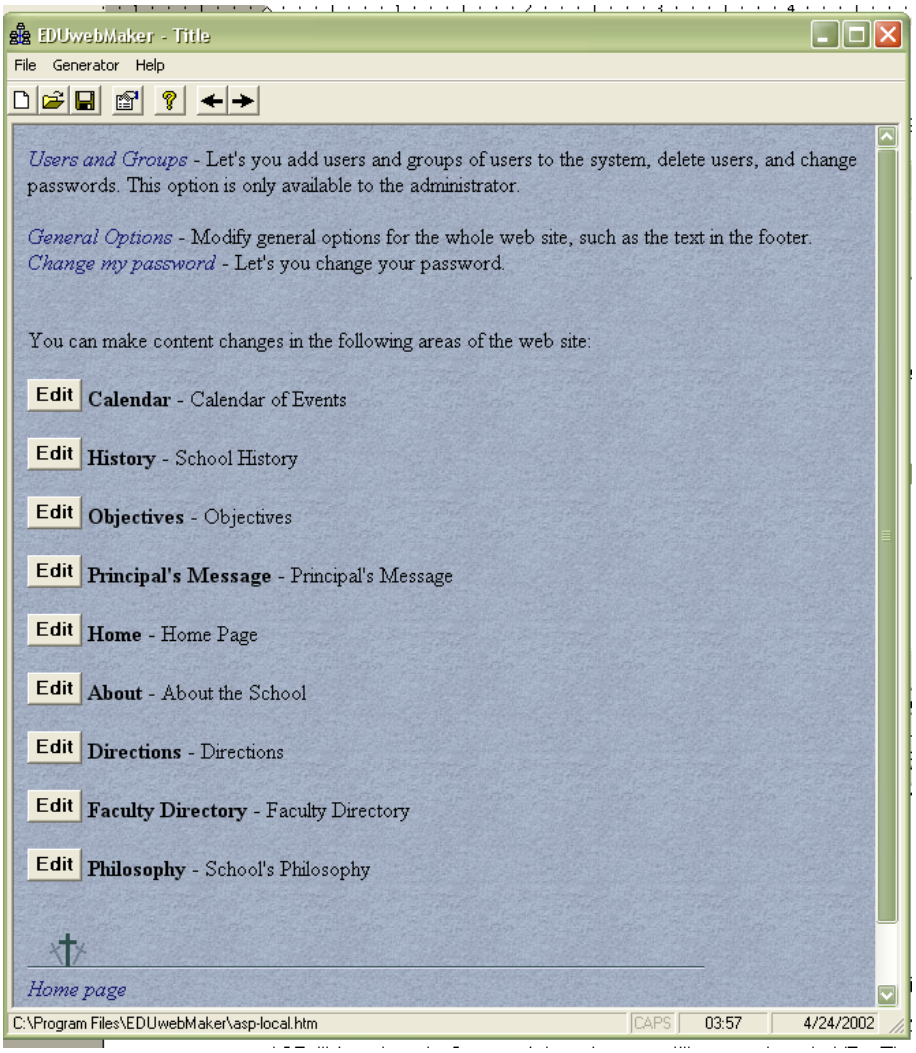


Figure 8 – Sample Administration Page

4.4 Program Architecture

EDUwebMaker was written as a Microsoft Visual Basic 6.0 application. It relies on three Dynamic Link Libraries (DLL): MyASP.dll, MyASP2.dll, and MyComp.dll. MyASP.dll is written in C++, and the other two dll's are written in VB. They are based on public-domain DLLs found on Microsoft's web site (http://www.microsoft.com). The program also relies on the following Microsoft-provided components (that are automatically

installed if they don't already exist when you install EDUwebMaker): Visual Basic controls, MS Internet Controls, MS ActiveX Data Objects (ADO) 2.5 Library, MS Scripting, MS Script Control 1.0, and MS HTML Object Library.

The EDUwebMaker VB project consists of the following components; note that all these files may be found on the CD that accompanies this report in the "Source Code" subdirectory:

Global Modules:

Module1.bas:	Contains startup code, global subroutines, and global variables and constants.

Class Modules:

CAspParser.cls:	Class that converts an ASP to HTML client-side.
Security.cls:	Class that defines the security model of the project.
Groups.cls:	Class that contains a list of CGroups.
CGroups.cls:	Class that defines the security of a group of users.
Users.cls:	Class that contains a list of CUsers.
CUsers.cls:	Class that defines the security of a user.
cProjInfo.cls:	Class that defines all the information about a project. Also serializes this information (saves and loads it from disk).
Features.cls:	Class that defines the features in a project. Part of cProjInfo.

Forms:

frmSplash.frm:	Splash screen that shows up while the program is loading.
frmMain.frm:	Main form. Contains the main user interface via menus and the HTML browser in the main part of the window.
frmProperties:	Allows the user to insert general information about their project and define the security model.

frmFeatures: Allows the user to choose the features they want for their web site.

frmAbout: Gives information about the EDUwebMaker program.

All these files contain VB code. Comments are used throughout the code. Each function is commented on what it does and how it interacts with other functions. In addition, there are comments within function to describe what exactly they are doing. See the source code on the CD to get an idea of how the program works.

In addition to the VB project, there are supporting files that are required for the project to work. First there is the setup.mdb database. This is a JET database. It has a table for each of the types of features for a web site (effects, features, layouts, navbars, and styles). In each table, each entry defines one feature that will be listed in the EDUwebMaker program. Each entry also defines where the files for the feature are located relative to the program directory. This database makes the EDUwebMaker program easily expandable (see below). A diagram of the structure of the database can be seen in Figure 6.

Next are the generic files that are used to build the final web site. These are located in the */data* directory. In general, each feature is defined by one or a few files under the data directory. These files are all independent of each other. This allows a programmer to add a new feature merely by creating a new file or two, and then entering their locations in the setup.mdb. Each type of feature (effects, features, layouts, navbars, and styles) is located in its corresponding subdirectory.

There are some general files located under the data directory. *aspstart.inc* and *aspend.inc* are generic include files that are included at the beginning and end of the ASP pages. They contain ASP commands to open the database and extract some general information from it. *content.mdb* is another JET database. It contains all the information for the website. It starts off blank. When a user creates a project, it is populated with project-specific data. In its "blank" state, it contains one table for each feature that the program supports, and default data for that web page. After the user generates the web site, they may use the administration section of the web site to add and modify the information that ends up in this database. Next, the *security.txt* file contains the definitions of the security models that may be selected by the user when setting up a project.

Finally, the *xbrowser.js* file is a JavaScript include file that gets included

setup.mdb

- ef fects
  - name
  - description
  - samplepic
  - filename
- features
  - name
  - description
  - samplepic
  - filename
  - level
  - simpleadmin
  - unique
- layout s
  - name
  - description
  - samplepic
  - filename
- navbars
  - name
  - description
  - samplepic
  - filename
- sty les
  - name
  - description
  - samplepic
  - cssfi le
  - backfile
  - dividerfile

Figure 9 – Structure of Setup.mdb Database

with the web pages. It causes some variables to be set to the current browser version and platform. It makes it easier for web pages to be cross-browser/platform compatible.

The files under the *admin* directory are the basic files for implementing the administration section of the web site. There are various include files that are included in all the administration pages. *authorize.inc* makes sure the user has logged on, in order to keep the administration section secure. *convert.inc* contains an encoding and decoding function that replaces a newline (carriage return/line feed) character with the HTML
. It also replaces double quotes with single quotes. This allows the program to display the text as it will appear to the user, without any HTML codes. The *default.inc* file contains the code necessary to open the database. Finally, *footer.inc* contains information that will be displayed at the bottom of each page.

There are numerous ASP pages under the *admin* directory a well. *default.asp* merely redirects the user to *login.asp*. *login.asp* allows the user to log in. It lets them log in with a username and password, and verifies it with the database. *main.asp* is the main page. Depending on the user's username, it displays whatever options that user has. *accounts.asp* and *accountsadmin.asp* allow a user to modify login accounts. *password.asp* and *updatepassword.asp* let a user change a their password. *startup.asp* changes the default startup page for the web site. *general.asp* let's the user modify some basic properties that affects the whole web site. The main page also links to all the web pages the user has administration rights too. These pages are defined under *features\admin*.

The files under features are divided into two subdirectories: admin and asp. Admin contains ASP pages (**.asp*) that are the administration pages for each feature (one page per feature). The ASP page is responsible for letting the user add, edit, and delete information for the given page. It retrieves and stores the information in the *content.mdb* file. Under the *asp* directory are located include files, one for each feature (web page). Each one contains the HTML and ASP code necessary for the bulk of each page. They access the database to get the information displayed on them.

The files under *effects* are include files (**.inc*) that contain JavaScript code that will give the web site a special effect. The files under *layouts* are ASP files that define the layout of all the pages in the web site. They contain the basic structure of the web page. They automatically include the *aspstart.inc* and *aspend.inc* files, the content file (from the *features\asp* directory mentioned above), and the navigation bar (mentioned

below), along with various values from the database. The *navbars* directory contains include files, one for each navigation bar style. They contain the HTML and ASP code necessary to display a navigation bar that will list and link to all the web pages in the web site. Finally, the *styles* directory contains all the files necessary to define how the web pages look. There are three subdirectories. *backgrounds* contains JPG graphic files for each of the supported background for the web site. *dividers* contains GIF graphic files for each of the supported divider graphics for the web site. Finally, *stylesheets* contains CSS (cascading style sheet) files for each style for the web site. These files describe the appearance of the web site, everything from font color and size to background image.

The final directory under the *data* directory is the *thumbnails* directory. It contains JPG images that are displayed in the program when the user is selecting web site properties. Each image gives a preview of what selecting that option will cause the web site to look like. For example, when the user clicks a feature, a sample of that web page is displayed. When a user clicks a navigation bar, a sample of it is displayed.

The final supporting files for the program are located in the *html* directory. These are various HTML files that contain help to the user on how to create a web site with the program. They can be viewed by going using the "Help" menu in the program, and are automatically displayed when necessary when working with the program.

Finally, installation files are built from all the program and data files so that the user may install the program on their system. The installation project and code were written in Visual Studio .NET and were generated automatically from a code "wizard". Along with the installation project are a couple support files. The *autorun.inf* file is placed in the root directory of the CD where the final installation files are placed. This file tells a computer that has AutoRun enabled which program to launch when the CD is put in the drive. It instructs the computer to run the installation program. Secondly, there is a *readme.txt* file. This ReadMe file contains all the information on how to install and run EDUwebMaker. It is installed with the program and placed in the root directory of the CD.

As you may see, there is a lot of complexity involved with this project. Not only is there the code for the actual program to worry about, but there is HTML and ASP code in dozens of support files, all referencing a dynamically-generated database that must be assembled together perfectly. However, in the end, handling all the complexity on the

programmer's end allows us to present a progra m that is simple for the user . The user has the power to create very complex web sites with minimal effort and complexity.

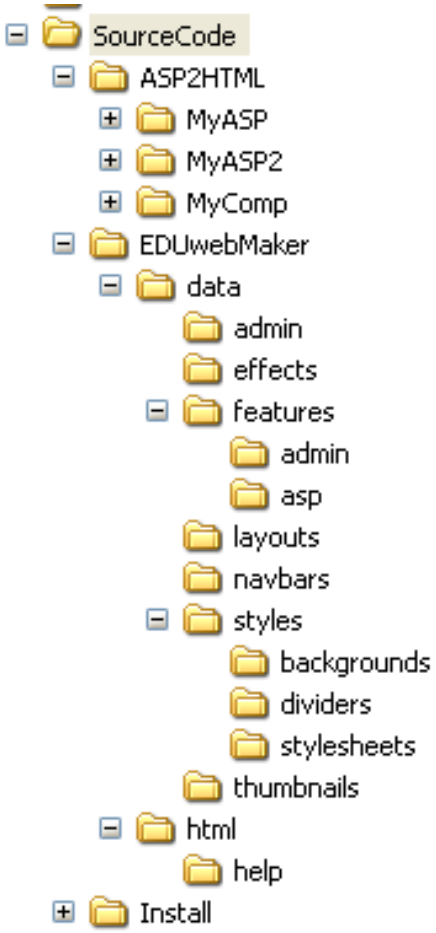


Figure 10 – Directory Structure of the EDUwebMaker Source Files

5. RESULTS

As mentioned before, our project was tested at three levels: function, functional, and user. At the most basic level, individual functions in the software were tested for functionality. This included checking algorithms such as the ones in the classes to make sure the objects performed their desired function. Many bugs were found during this phase and corrected.

The project was also tested at the functionality level. Here, the software was tested to see how well the specified requirements, listed in section 4, were met. Basically, the software had to work as designed. The installation was tested on different operating systems and was found to work on Microsoft Windows2000 and later; however, it was a little quirky on earlier operating systems. Otherwise, it performed all its required functions adequately, including being able to create, save, and close projects, generate web sites, view those web sites after creation, and allow the user to administer them.

Finally, the project underwent user testing. Since we knew our software prototype now worked, it was time to see if it achieved its more qualitative goals. Those would be power, flexibility, ease of use, and security. To perform the user testing, we were able to demonstrate our software, specifically the actual program and not such things as the installation or the overall unfinished look, to two school employees and get their feedback. The two employees were Carol White, a Special Education Assistant at Hampstead Middle School in Hampstead, NH, and Lynne Croswell, a Substitute Teacher on the Board of Directors at Christian Heritage School in Trumbull, CT. Here are the questions asked and their responses after they tested the software.

1. Do you think that there would be someone at most schools who could run our program and generate a website?

Mrs. White: Yes, most schools have a technology person who keeps the computer systems running. They would easily be able to do the initial set up of this software.

Mrs. Croswell: A small school may not have the tech support as far as staff designation but could hire a consultant if someone in-house did not have the capability to generate the site. With today's technological climate it is hard to imagine that there would be no one who could do this, even if it was a parent.

2. Do you think that the average teacher could administer the web site once it has been generated?

Mrs. White: Yes, though it would be helpful if the technology person would run an in-service training session so everyone is familiar with what the software can and cannot do. Teachers might have to be periodically reminded to keep their web site up-to-date with current information.

Mrs. Croswell: I am uncertain about the capabilities of the "average teacher" but I think that certainly one who is competent in basic computer skills could administer the site once it is created.

3. Could the web sites generated by this program satisfy the needs for a school web site?

Mrs. White: Absolutely, though it would be nice if pictures could easily be added to the web site. Also, being able to e-mail the school with questions is a nice plus.

Mrs. Croswell: I would think so. Did you have a feature that would allow the administrator to create and format custom pages or would they only be able to choose from your options?

4. What features did you like about the program?

Mrs. White: When setting up the web site the choices were limited. If one is given too many creative options, one can get bogged down in trying to make it look nice and therefore, it becomes too time consuming to be updated on a regular basis. The web pages offered were clear and easy to read and understand once the information had been entered onto them.

Mrs. Croswell: I like the fact that the site can be accessed through any internet connection, rather than one main computer or a school network computer.

5. What features did you dislike about the program?

Mrs. White: I found the explanations about how to set up the web site to be too long and complicated. The directions need to be simplified for the non-technical person.

Mrs. Croswell: I would love to see the design options you would include and ways to give the sight a personal look. (I think the Christian Heritage School site I mentioned at the top is pretty bare-bones and not terribly creative and appealing. Perhaps that is the norm though and I expect too much.)

6. The program gave you a choice of what kinds of web pages you wanted on your web site (e.g. calendar page, home page, faculty directory page, etc.). What additional kinds of web pages do you think would be useful for the program to be able to generate?

Mrs. White: A page for sports/chorus/play etc practices. The regular calendar would get too crowded with all those dates. Perhaps each dept. might want their own calendar page.

Mrs. Croswell: I would love to see pages for the lunch menu, fine art samples which could also include text as to who did the work, student of the month/ outstanding student feature, sports schedule and record update, alumni, outstanding faculty recognition/

achievement, and volunteer opportunities and/or recognition. These are what I thought of off the top of my head.

6. ANALYSIS OF RESULTS

We received responses from both of the teachers who tested the software. Overall, the responses were pretty favorable and supported the need and usefulness of a the type of product this prototype demonstrates. Since the software is just a proof of concept or a prototype there was limited functionality and features but enough to test anyway. An analysis of the answers is below.

Both teachers thought that at most schools there would be somebody, perhaps a parent at worst, who would be capable of installing and setting up the software. Mrs. Croswell suggested that possibly a school could bring in a consultant to at least set up the software, but they both said that the school should have someone experienced with computers especially nowadays with computers being so important in daily life. Also, they both agreed that as long as a teacher had basic computer skills and possibly a short in-service training session on how to use the software, it would be quick and easy for one to update and administer one's portion of the website. Again, the teachers were in agreement that the EDUweb Maker could generate a website which was feature rich and useful so that it would satisfy the needs of the school. A feature wch Mrs. White liked was the limited choices in customizing the look of a particular page to keep things simple and uncomplicated. Both teachers liked being able to administer the website from any computer in the world with a connection to the internet. The features which they didn't like were the rather long and complicated explanations and the lack of options for customization to personalize a schools website. These features, of course, would be incorporated in a finished version of the software, perhaps as another project, and not in this prototype. Other types of pages in the website which the program doesn't currently incorporate and the teachers would like to see include, separate pages for sports, chorus, and play practices to keep the main calendar page uncluttered, fine arts samples, lunch menus, outstanding student features, alumni, volunteer opportunities and achievement recognition pages. Since the school would likely have a staff member capable of operating the software or could at least have a parent help out, the cost of using this program would only include the hosting fees from an internet service provider which can range from $200 to $1200 per year [4]. The benefit to cost ratio would be very good. However, hiring a design firm to make a quality site for a school would severely lower the benefit to cost ratio. A high quality website which costs $1000 or less because of a host-

ing fee is much better than a high quality website which costs thousands of dollars just to develop and maintain then even more to host the website as well.

7. CONCLUSIONS AND RECOMMENDATIONS

Through the course of this project we determined a need for an inexpensiuve and easy way to create feature rich and useful websites for schools without resources to make one using current methods. The EduWebMaker prototype demonstrates some of the functionality and usefulness of a program which would aid in making such a website. After having the software tested by some teachers and analyzing the feedback which they provided, we have found the software to be useful to schools for providing a simple to use development platform through which a feature rich and effective web presence for a school can be created and administered. By making use of this program a school can have an effective way of communicating between teachers, students, parents, and outsiders. Some recommendations for future work would include adding more features and customization options since the prototype is rather bare-bones in its current state. With some more time and effort (i.e. a company making it into a finished product) this unique program could be made very powerful and extremely useful to schools without resources for much computer technology and even possibly to schools which already employ technology experts.

REFERENCES

1. CompUSA. (Retrieved 4-24-2002 from http://www.compusa.com).

2. Association of Christian Schools Inter national. (Retrieved 4-24-2002 fro m http://www.acsi.org/).

3. Resource Guide to Federeal Funding for Technology in Educatioin. U.S. Department of Education. (Retrieved 4-24-2002 from http://www.ed.gov/Technology/tec-guid.html).

4. 2001's Median Prices For Full-Site Develo pment. BtoB. (Retrie ved 4-24-2002 fro m http://www.btobonline.com/webPriceIndex/index.html).

5. Hosting With ASP Support. iHostcafe.com. (Retrieved 4-24-2002 from http://ihostcafe.com/search/asp.php3?pkdev=Active%20Server%20Pages).

APPENDICES

Appendix A – Survey of Schools

- Do you currently have a web page for your school?

- If you were going to create a web site for your school, which of the following features would you want to include on it?

  - ❑ Message on homepage if school is delayed, canceled, etc.
  - ❑ General Info
  - ❑ Objectives
  - ❑ Philosophy
  - ❑ History
  - ❑ School facts
  - ❑ Tuition
  - ❑ Financial Aid
  - ❑ School songs, etc.
  - ❑ Student Handbook/rules
  - ❑ Directions
  - ❑ Maps
  - ❑ Other _____
  - ❑ News
  - ❑ Message from the principal

- ❑ News items
- ❑ Archive of news items
- ❑ Other _____
- ❑ Calendar (different formats)
- ❑ Students
- ❑ Directory
- ❑ Optional home page link
- ❑ Parents
- ❑ Alumni
- ❑ News
- ❑ Reunions
- ❑ Faculty (teachers and administration)
- ❑ Directory
- ❑ Bio
- ❑ Home page link
- ❑ Class links
- ❑ Teacher of the year
- ❑ Other _____
- ❑ Academics
- ❑ Curriculum
- ❑ Class descriptions
- ❑ Course handbook
- ❑ Divided into K, 1-6, Jr High, High School

- ❑ Areas: Science, art, etc.
- ❑ Courses
- ❑ Info
- ❑ Description
- ❑ Course web page
- ❑ Summer programs
- ❑ Other _____
- ❑ Athletics
- ❑ News
- ❑ Calendar
- ❑ Schedule
- ❑ Scores
- ❑ Coaches
- ❑ Individual Sports
- ❑ Team rosters
- ❑ Coaches
- ❑ Scores
- ❑ Schedule
- ❑ Other _____
- ❑ Activities
- ❑ Clubs/Organizations (info web page or custom web page link)
- ❑ Community Service
- ❑ Other _____

- ❑ Facilities/Services
- ❑ Library
- ❑ Link to card catalog
- ❑ Computers
- ❑ Computer use policy
- ❑ Guidance
- ❑ College Guidance
- ❑ Other _____
- ❑ Publications
- ❑ Newsletter
- ❑ Current
- ❑ Archive
- ❑ Yearbook
- ❑ Other _____
- ❑ Community links
- ❑ Links
- ❑ Site index
- ❑ Additional "custom" web pages can be added at any level

- What additional features would you want on the site?

- Would you create/update a web site for your school if you could have any of the features mentioned in this survey on the site?

- Would you like information in the future about this project when it is completed?

Thank you for filling out the survey. Please return it in the enclosed envelop.

Appendix B – Results of Survey

Academics:

ID	Curriculum	ClassDesc	CourseHandbook	Courses	CourseWebPage	Summer	Other	OtherVal
0	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	
1	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	
2	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	
5	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	
6	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	
7	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	
8	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	
9	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	
10	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	
7		5	5	6	4	4	1	

Activities:

ID	Clubs	CommServ	Other	OtherVal
0	FALSE	TRUE	FALSE	
1	TRUE	TRUE	FALSE	
2	FALSE	FALSE	FALSE	
5	TRUE	TRUE	FALSE	
6	TRUE	TRUE	FALSE	
7	FALSE	FALSE	FALSE	
8	FALSE	TRUE	FALSE	
9	TRUE	TRUE	FALSE	
10	TRUE	TRUE	FALSE	
5		7	0	

Athletics:

ID	News	Calendar	Schedule	Scores	Coaches	Individual	IndRosters	Other	OtherVal
0	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	
1	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	
2	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	
5	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	
6	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	
7	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	
8	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	
9	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	
10	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	
	8	8	8	8	7	6	5	0	

Alumni:

ID	News	Reunion	Other	OtherVal
0	FALSE	FALSE	FALSE	
1	TRUE	TRUE	FALSE	
2	TRUE	TRUE	TRUE	alumni email addresses
5	TRUE	TRUE	FALSE	
6	TRUE	TRUE	TRUE	Alumni can e-mail their addresses and e-mail addresses to school
7	FALSE	FALSE	FALSE	
8	TRUE	TRUE	FALSE	
9	TRUE	TRUE	TRUE	
10	TRUE	TRUE	FALSE	
7		7	3	

Facilities:

ID	Library	LibLinks	Comps	CUP	Guidance	CollegeGuidance	Other	OtherVal
0	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	
1	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	
2	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	
5	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	
6	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	
7	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	
8	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	
9	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	
10	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	FALSE	
	6	4	6	5	6	6	0	

Faculty:

ID	Directory	Bio	Homepage	ClassLinks	TOY	Other	OtherVal
0	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	
1	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	
2	TRUE	FALSE	TRUE	TRUE	FALSE	FALSE	
5	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	
6	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	
7	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	
8	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	
9	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	
10	TRUE	FALSE	FALSE	TRUE	TRUE	FALSE	
7		4	3	5	2	0	

General Info:

ResponseID	Objectives	Philosophy	History	Fact s	Tuition
0	TRUE	TRUE	FALSE	TRUE	TRUE
1	TRUE	TRUE	FALSE	TRUE	TRUE
2	TRUE	TRUE	TRUE	TRUE	FALSE
5	F ALSE	TRUE	TRUE	TRUE	FALSE
6	TRUE	TRUE	TRUE	TRUE	TRUE
7	FALSE	FALSE	FALSE	F ALSE	F ALSE
8	TRUE	TRUE	TRUE	TRUE	FALSE
9	TRUE	TRUE	TRUE	TRUE	TRUE
10	TRUE	TRUE	TRUE	TRUE	TRUE
7		8	6	8	5

Aid	Songs	Handbook	Directions	Maps	Other	OtherVal
TRUE	FALSE	TRUE	TRUE	FALSE	F ALSE	
FALSE	TRUE	TRUE	FALSE	TRUE	F ALSE	
FALSE	F ALSE	FALSE	TRUE	TRUE	FALSE	
FALSE	FALSE	F ALSE	FALSE	FALSE	F ALSE	
TRUE	F ALSE	TRUE	TRUE	TRUE	FALSE	
FALSE	FALSE	F ALSE	FALSE	FALSE	F ALSE	
FALSE	F ALSE	TRUE	TRUE	TRUE	FALSE	
TRUE	F ALSE	TRUE	TRUE	TRUE	FALSE	
TRUE	TRUE	TRUE	TRUE	TRUE	F ALSE	
4	2	6	6	6	0	

News:

ResponseID	Message	News Ar	chive	Other O	therVal
0	TRUE	TRUE	FALSE	FALSE	
1	F ALSE	TRUE	TRUE	FALSE	
2	TRUE	TRUE	FALSE	FALSE	
5	F ALSE	TRUE	FALSE	FALSE	
6	F ALSE	TRUE	TRUE	FALSE	
7	F ALSE	FALSE	FALSE	FALSE	
8	F ALSE	FALSE	FALSE	FALSE	
9	TRUE	TRUE	TRUE	FALSE	
10	TRUE	TRUE	TRUE	FALSE	
4		7	4	0	

Publications:

ResponseID	Ne wsletter	NewsCurr	NewsArchive	Yearbook	Other	OtherVal
0	TRUE	TRUE	TRUE	FALSE	FALSE	
1	TRUE	TRUE	TRUE	TRUE	FALSE	Student News-paper
2	F ALSE	TRUE	FALSE	FALSE	FALSE	
5	TRUE	TRUE	FALSE	FALSE	FALSE	
6	TRUE	TRUE	TRUE	FALSE	FALSE	
7	F ALSE	FALSE	FALSE	FALSE	FALSE	
8	F ALSE	FALSE	FALSE	FALSE	FALSE	
9	TRUE	TRUE	TRUE	FALSE	FALSE	
10	TRUE	TRUE	TRUE	FALSE	FALSE	
6		7	5	1	0	

Students:

ResponseID	Dire ctory	Homepage	Other	OtherVal
0	F ALSE	FALSE	FALSE	
1	F ALSE	FALSE	FALSE	
2	F ALSE	TRUE	FALSE	
5	F ALSE	FALSE	FALSE	
6	TRUE	FALSE	FALSE	Note that we'd love to include a student section on our current web school
7	F ALSE	FALSE	FALSE	
8	F ALSE	FALSE	FALSE	
9	F ALSE	FALSE	FALSE	
10	F ALSE	FALSE	FALSE	
1		1	0	

Appendix C – Source Code for EDUwebMaker

The following files may be found on the CD which our project is on. The following is the locations where the files may be found.

Installation Files: \Source Code\EDUwebMaker

Appendix D – Source Code for ASP2HTML

These files were taken from the Microsoft website:
http://msdn.microsoft.com/msdnmag/issues/0900/cutting/cutting0900.asp
They are public domain. They were modified in order to provide the needed features for the program. You may find the full source code on the CD with our project on it.

Appendix E – Data Files

The following files may be found on the CD which our project is on. The following is the locations where the files may be found.

Installation Files: \Source Code\Install

Databases: \Source Code\EDUwebMaker
\Source Code\EDUwebMaker\Data

Data Files: \Source Code\EDUwebMaker\Data

Appendix F – Help Files

welcome.htm:

Welcome to EDUwebMaker

Welcome. If you would like to load a project you were previously working on, go to the *File* menu and select *Open*.

To start a new project, go to the *File* menu and select *New*. Select a location and a name of your project. The program will create a Project File with all your selected options. When you generate your web site, it will get placed in a sub-directory from the location you selected.

Next, a preferences window will pop up (you get always get back to this window later by going to the *File* menu and selecting *Properties*). On the first page you can set basic information about the site, like the name of the project and the name of the school.

On the second page you can select the security model for the web site. This is an important step. One of the features of your web site will be that you can update the content from any browser. However, to make sure unauthorized people don't change your web site, you must set up accounts for everyone that is allowed to edit your web site. Then you can select what pages each user can edit. Users may also be organized into groups. For example, you could have an "admin" account that lets you change anything, a "teachers" group containing individual teacher names for each teacher to allow them to update their own web pages, and a "sports" account to allow the PE teacher to update all the sports pages. You may either select a predefined security model, or create your own. Make sure to set passwords for each user. Note that you will be able to change the security model later, even after you're web site it up on the internet.

Go to help contents.

newproj.htm:

How to Set up your Web Site

Now that you've created your new project, it's time to select what's going to go into this web site. You may want to just play around with many of the options and try out the resulting web page before you make a final decision.

- Go to the *Generator* menu and select *Select Features*.

- You'll be presented with five pages of options:
  - **Layout** - Select a layout for your web site. This is the general format of the site.
  - **Navigation** - Select a method to allow the people that visit your site to navigate through it. These are normally different kinds of navigation bars.
  - **Style** - Select a style to set the look and feel of your site. Styles include the background texture, the colors of the text and links, and text formatting.
  - **Features** - This is the big one. Select which features (content) your web site will contain. Each of these features will turn into a separate web page on your site.
  - **Effects** - Select any additional effects for your web site to make it more interesting or useful. Effects will be displayed on every page.

Back to help contents.

features.htm:

How to Generate a Web Site

Now that you've selected all the options for your web site, you need to generate it. First, however, you should save your changes.

- Go to the *File* menu and select *Save*.
- Then, to generate the web site, go to the *Generator* menu and select *Generate Web Site*.

Back to help contents.

generate.htm:

How to Test the Web Site

Now that you've generated the web site, it is time to test it.

- Then, to test the web site, go to the *Generator* menu and select *Test Web Site*.

If you don't like it or want to try something different, just go back to *Select Features*, select new options, and then generate the web site again.

If you like the web site, then start adding your specific content to it. There will be a link somewhere on your Home Page that will say something like: "Click here to administer the web site". Click this link. You are now in the website administration area. You may log on as any of the Users or Groups you specified before, or "admin" (with password "admin") to login as Administrator. As Administrator, you can do everything. You will probably want to change the password for administrator as soon as possible. For more information on Users, Groups, and passwords, see Administration help under the Help menu.

Once you've logged in, you may click on any of the listed web pages (features) to insert or change the information on that page. For more information on how to administer the web site, see the help file.

After you've entered all the information into the web site, and you're ready to get it up on the internet, you need to find a way to "host" your site. For detailed information on how to get your web site up on the internet, see the Hosting help under the Help menu.

Back to help contents.

admin.htm:

How to Administer Your Web Site

Once you have generated your web site, you may administer it. Administering your web site involves updating the information on it, or updating the security model. You can administer your web site while you're still testing it by going to the *Generator* menu and clicking *Administer Web Site*.
You can also administer the web site after it has been hosted. You do this to update the data on the web site so the web site is always up-to-date. You can do this by clicking the link at the bottom of the home page that says administer web page (if your home page has it). Alternatively, you can browse to **/admin/** on your web site (e.g. http://www.yourwebsite.edu/admin/).

Once you get to the initial administration page, you will be asked to log in. This is so that other people can't hack into the site and change your web pages. You may enter any of the username/password combinations defined in the security model, and you will be given the permissions defined in the security model. After this, the main administration page that gives you all your options loads. Depending on your permissions, you can change security model settings (add/remove groups and users, change their permissions, and change their passwords), change your own password, or make changes to specific web pages.

Back to help contents.

hosting.htm:

How to Host Your Web Site

If you have generated, tested, and added all the data to your web site, then you are ready to get it hosted. Once it is hosted, other people will be able to get to it. You can either host the web site yourself or get a company to host it for you.

- **Hosting the web site yourself:**
  If you are planning on hosting the web site yourself, you will at minimum need a computer that is always on and connected to the internet, that has Microsoft IIS on it, and you'll need your own domain name. If any of that just confused you then you need to get your school's technical person, or you'll need to have your web site hosted by a company.

  To host the site on your own computer, get the computer set up to serve web pages, and then upload the contents of your project directory to the root directory on the web server. The web site should display fine with the default settings of Microsoft IIS.

- **Having a company host your web site:**
  To do this, you will need to find a web hosting company (there are many out there) that offers you the ability to serve ASP (Active Server Pages). They should also provide you with a domain name (e.g. www.yourschoolname.com). Once you find the one that meets your needs, just sign up with them and send them the contents of your project directory. They should be able to do the rest.

Back to help contents.