

## **Building A Web Based System for Automated Grading of Computer Modeling Assignments.**

**Bob Fithen**

**Arkansas Tech University**

### **Abstract**

Arkansas Tech University like most Engineering programs requires a computer programming and application course. One major challenge in handling such a course is reflected in the grading methodologies for the assignments. Simple and loosely specified assignments by the professor lead to unorganized responses by a majority of students. In contrast, highly specific problem statements give the professor the flexibility to use these specifications to determine the grade on a particular assignment. The major problem with this approach is twofold. First, the professor must write a detailed specification for each assignment. Second, the professor must carefully assess each assignment to determine if in fact the student satisfied the given specifications. The second item requires the most time and effort. Traditionally students simply handed in printouts of their computer codes, outputs and some descriptive comments. As a result of the written presentation skills of some students, large amounts of time is required to determine if the given specifications were actually met. In response to this issue the author has written a dynamic web based system that automatically grades each homework assignment. Each assignment will be grade entirely based on meeting the given specifications. For each assignment a grading code must be written which meticulously checks each submission for correctness. Since each assignment may have multiple specifications, partial credit is possible for those multi-part assignments. When each student submits his or her assignment the code is stored in a predetermined location on the web-server and the grading code is automatically launched. The student is given immediate feedback through a set of diagnostic messages. In addition the student's grade for each assignment is immediately stored on the web-server and the student may check his or her grade on their own individual web page.

### **1. Introduction**

This paper describes a web-based system built by the author that accepts, grades, and tracks student's progress through an engineering based computer-modeling course. This computer-modeling course uses Matlab as is central programming language.

Traditionally students are given assignments consisting of problem statements, code requirements and presentation specifications. Upon completing these assignments the professor is left with the task of determining if the algorithm itself is correct. In addition the professor must determine if the student has properly understood the engineering behind the algorithm. The professor can determine the student's comprehension by observing written description, plots, data outputs from the assignments, etc. In this “*one-pass assignment collection*” method the student must complete an algorithm using Matlab and use this algorithm to develop plots and write interpretations of his or her results. The correctness of student's plot and written descriptions are contingent on writing a correct algorithm. Students tend to concentrate on either the algorithm itself or the interpretation of the results. However, concentrating on the results produced from an incorrect algorithm will yield the entire assignment incorrect. For this reason, the author began a project which grades the algorithm prior to any interpretations written by the student. In essence, this approach can be thought of as a “*two-pass assignment collection*” method.

## 2. System Description

The “*two-pass assignment collection*” method is very well suited to web development. The first pass has as its focal point, algorithm development. The process begins by students reading a web page containing a complete specification of the algorithm. The web page also contains a form available for students to submit their algorithms. Upon submittal of their Matlab program, a perl<sup>1,2</sup> script on the web server will take in and save their submission. This perl script will then launch a shell script, which in turn will launch Matlab. The Matlab code will load a grading script calling the students submission as a subroutine and test their routine against the given specifications. After completing this process a grade associated with the algorithm development is recorded and available for inspection by the student. Part of a typical web page specification is shown in figure 1.

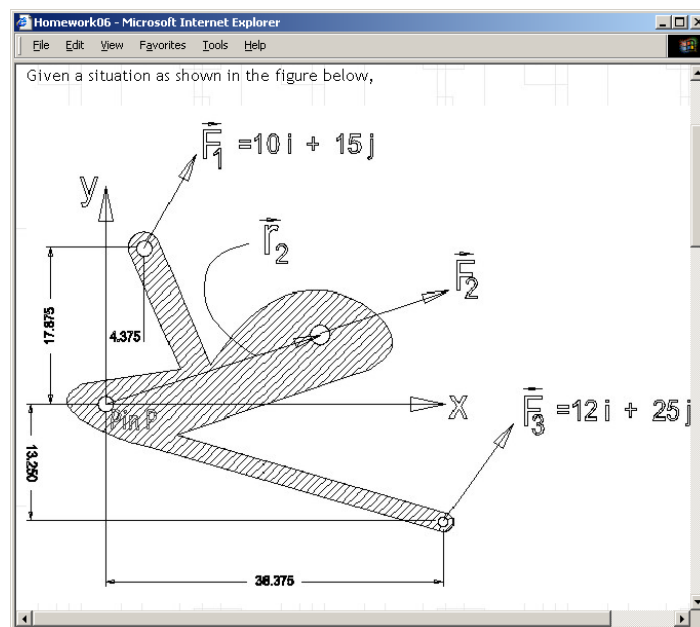


Figure 1. Graphic problem statement.

Homework06 - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites History Mail Print

Address [https://mengr.atu.edu/cgi-bin/Web\\_Testing/Process\\_Page.cgi?file=homework06.htm](https://mengr.atu.edu/cgi-bin/Web_Testing/Process_Page.cgi?file=homework06.htm) Go

Write a function subroutine to do the following;

■ Given  $\vec{r}_2$  and  $\vec{F}_2$  and Vmag;

1. Return a scalar value of -1000 if both  $\vec{r}_2 = \vec{F}_2 = 0$
2. Return a scalar value of -2000 if both vectors  $\vec{r}_2$  and  $\vec{F}_2$  are nonzero.
3. if (given a nonzero value for the vector  $\vec{F}_2$ ) then
  - return  $\vec{r}_2$  such that the moment about the pin P is zero.
4. if (given a nonzero value for the vector  $\vec{r}_2$ ) then
  - return  $\vec{F}_2$  such that the moment about the pin P is zero.
5. Return a scalar value of -3000 if the given combination is not possible.

**NOTE: Vmag is the magnitude of the vector in question.**

function Rvalue=Homework06(F2,r2,Vmag)

Rvalue=

Submit


Figure 2. Written problem specification.

This figure contains a picture associated with the assignment. The actual problem statement with specification is shown in figure 2.

The “two-pass assignment collection” method allows the student to concentrate first on meeting the algorithm goals of the assignment. This process also relieves the professor from the tedious task of determining whether the student accomplished the algorithm correctly. Once the algorithm goal has been met, the student may progress with full knowledge that his or her Matlab program is correct. This confidence will allow student to concentrate on the results given by their Matlab code. Students will alter input data in order to observe how particular output variables change. This process will allow student to learn the physics behind the each example.

### 3. Web Server Description

The web system is developed around a Linux Mandrake system. However, the web system was an additional component using OpenSSL<sup>3</sup> and SSLmod<sup>4</sup> operating under the Apache<sup>5</sup> web server. All three software packages are available freely on the web. This secure system was used primarily for protection of the student as well as protection of the

professor's course material. Around this web server the entire system works within the context of the perl programming language and Microsoft's FrontPage. Once a professor develops a FrontPage web, the directory structure is stored in a location on the Linux machine not directly accessible by the web server. All files and directories contained within the FrontPage web must be readable by "nobody" since all web based cgi codes are run as user "nobody". Students enter the web system by visiting the web page shown in figure 3. Notice the lock, , at the bottom of this web page. This symbol indicates a secure connection between the web browser and the web server thereby minimizing the possibility of intrusion into the communication.

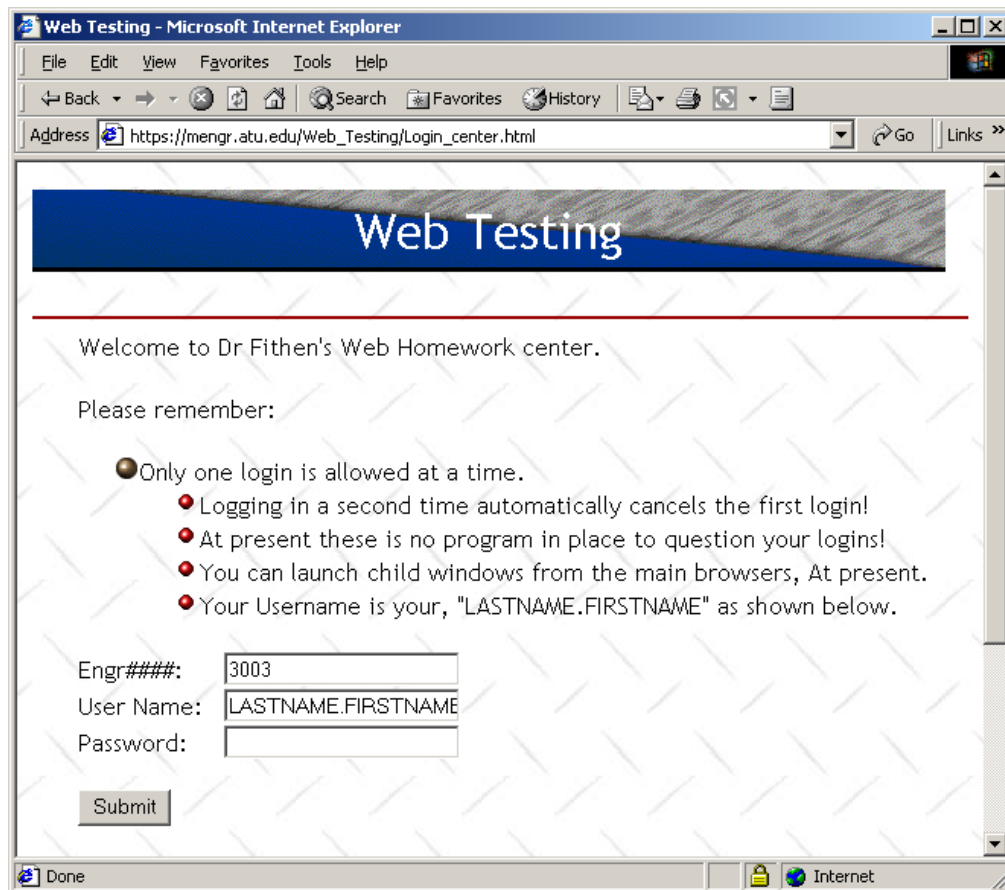
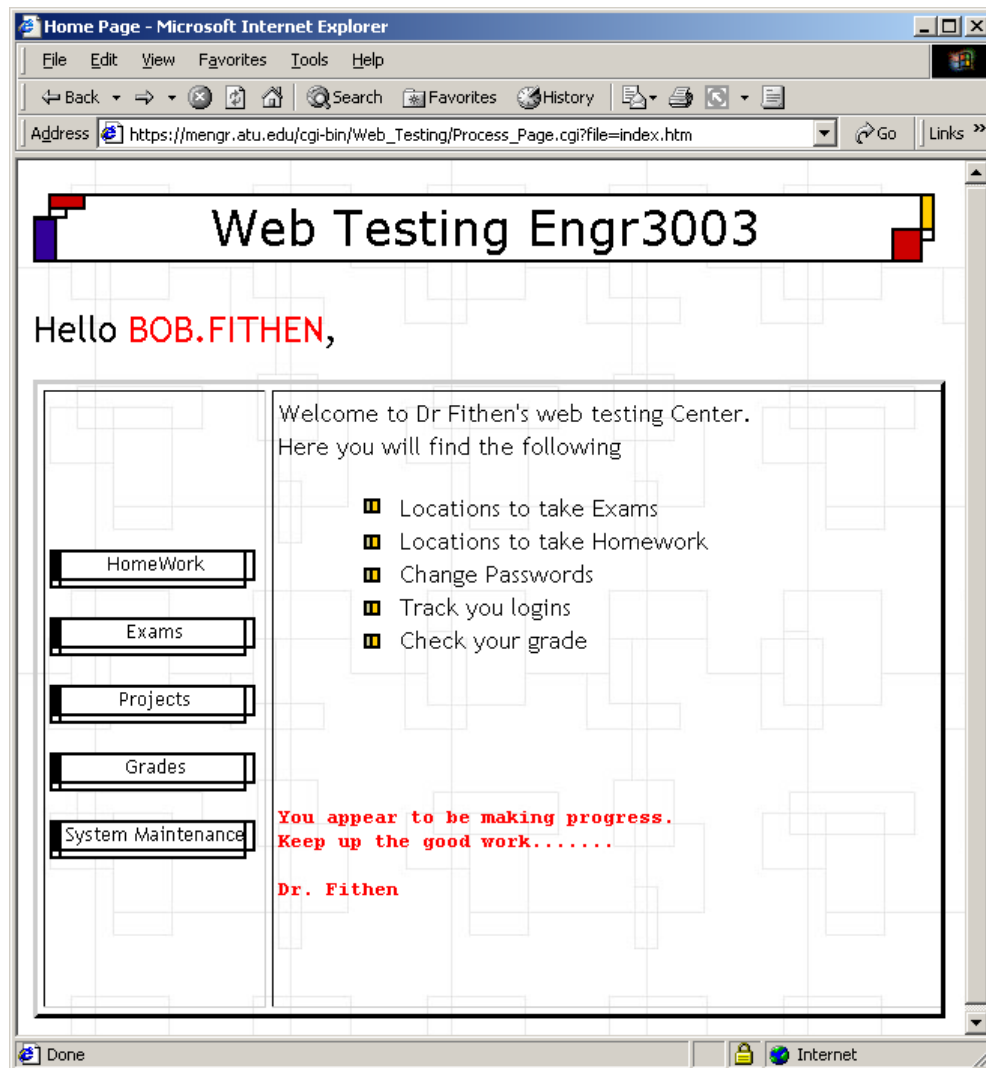


Figure 3. Secure web page login.

Upon entering the correct User Name, Password, and course number a perl code is executed on the server when the submit button is hit. This perl code will attempt to set three per-session cookies available for retrieval in the directory [https://mengr.atu.edu/cgi-bin/Web\\_Testing/](https://mengr.atu.edu/cgi-bin/Web_Testing/). These three cookies are username, engineering course number and an encrypted version of the password. The password is encrypted via the crypt command. These three cookies will allow the user to maneuver around within the web site without reentering their information. These cookies will expire upon closing the browser; hence no one can access the system by re-launching the browser and searching through the history. After entering the correct information the browser is redirected to the main location as shown in figure 4.



**Figure 4. Introduction screen.**

Five major areas are available to the student as shown in Figure 4. The “Homework” section contains all the homework assigned to the student during the course including locations to submit each assignment. The “Exams” area contains all exams given throughout the course including on-line as well as paper exams in pdf format. The “Projects” area contains a location for large homework problems. This section was not used during the Fall 2000 semester. The “Grades” area contains a dynamic page, listing grades for the student. When processing this page the perl program will replace a predefined text screen with the grade for that assignment. An example of the “Grades” page is shown in Figure 5. All pages are processed through a perl program “Process\_Page.cgi”. This perl program is capable of delivering html, images, and pdf formatted pages.

any of the grades are not in agreement with your completed assignment  
please let Dr. Fithen know.

Assignment	Your Score	Possible Points	Due Date
homework01	10	10	Sep 30
homework02	10	10	Sep 30
homework03	10	10	Sep 30
homework04	10	10	Sep 30
homework05	15	15	Sep 30+x
homework06	0	70	Sep 30+x
homework07	45	45	Sep 30+x
homework08	30	90	Sep 30+x
homework09	0	30	Sep 30+x

**Figure 5. Grade section of the web system.**

#### 4. Students Data

One key to success of the system is its ability to reduce the bookkeeping required by the professor. To this end, each student is given his or her own location in the directory structure. This location will be class dependent and therefore the top-level directory will be, for example, *system\_root/ENGR3003*. Users directories will be located in, for example, *system\_root/ENGR3003/LASTNAME.FIRSTNAME*. In each students directory is a complete list of past submissions, grades, session id index, password, a specialized message tailored for each student, and a log file containing information about each login. In addition a directory in the structure contains the solution and a grading code for each assignment. One of the most powerful features of this system is its ability to give partial credit for subtask within the specifications, figure 1. In the example shown in figure 1, each numbered item 1 through 5 carries a different number of points and is graded individually. This allows the professor to divide the task into subparts for the student.

## 5. Experiences

The author began writing this web-based system on September 1, 2000. This in itself was a challenge primarily due to students submitting assignments within a few minutes after the author completed the grading code. With the exception of a few insignificant problems, the system performed very well. These minor problems were primarily due to the grading code being written too stringent. One example of this case involved the solution of two simultaneous second order differential equations that simulate a low pass filter. A transient response is required for time between zero and 0.02 seconds. The author wrote a grading code that compared a cubic spline fit between the correct solution and the student's submission. By setting the difference between these two functions to stringent the web-system returned, to the student, an error when in fact their code was passable. Because of a quick correction, this problem remained on the system only a few hours. One other problem involved the ability of students to submit endless loops. This problem was handled by writing a perl program that constantly check the status of any Matlab process run by user "nobody". When these process run beyond a predefined time the perl code will "kill" this process.

Student's response to the system has been very favorable. Items in favor include; the ability to submit their assignment over and over without penalty; the immediate feedback from the web by posting the grades; The ability to submit assignments 24 hours a day 7 days a week. Negatives include, insufficient diagnostic messages during a failed attempt at submitting.

## 6. Conclusions

This paper described a web-based system built by the author that accepts, grades, and tracks student's progress through a computer-modeling course. The computer-modeling course uses Matlab as is central programming language. For this reason the major focus of this effort is the implementation and testing of a system which takes in students homework assignments through a web page, grades their assignments and records their grades in a web based grade book. In order to protect the student's privacy as well as the privacy of the course content, the web pages are delivered through a secure connection. Through this secure web page, the author has written a server-based application interface using the Perl language. Through this interface the students are required to enter a username and password to enter the site, thereby limiting access to the site to the intended audience. Upon entering the site students can observe their grades, submit homework assignments, submit projects, and take exams. The web-server runs Linux with Apache as its web server. Apache is configured to operate on port 443 using Secure Socket Layer (SSL) as it communication pipe. At the core of the entire web system is a Linux version of Matlab. When students submit an assignment through the web, the Perl interface will save their submission as a file and launch Matlab. Matlab is instructed to run a grading script that in turn will call their routine. The grading script is written in Matlab script by the professor and will grade the student's submission based on the given set of software specifications known to the student. Since the specification may include more that one item, partial credit can be given on any assignment.

## Acknowledgments

The author wish to thank NSF for supporting this project under NSF project #9952284

## Bibliography

1. Clinton Pierce, *Teach Yourself Perl in 24 Hours*, SAMS, 1999.
2. Larry Wall, Tom Christianson and Jon Orwant, *Programming Perl*, 3<sup>rd</sup> edition, OReilly
3. Open SSL Web Site: <http://www.openssl.org>
4. ModSSL Web Site: <http://www.modssl.org>
5. Apache Web Site: <http://www.apache.org>

## **BOB FITHEN**

Bob Fithen is an assistant professor at Arkansas Tech University. He received his B.S. in Mechanical Engineering from Louisiana Tech University, M.S. in Mechanical Engineering from Texas A&M University, and his PhD in Engineering Mechanics from Virginia Tech University. He spent four years working at General Dynamics, Fort Worth and a total of five years working in the research division of Wright Laboratories in Dayton, Ohio. Further information may be obtained at <http://mengr.atu.edu>