COMPREHENSIVE WEB APPLICATION FOR CITER

A Thesis
Presented to the
Faculty of
San Diego State University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Shivam Dixit
Spring 2015
SAN DIEGO STATE UNIVERSITY

The Undersigned Faculty Committee Approves the

Thesis of Shivam Dixit:

Comprehensive Web Application for CITER

Carl Eckberg, Chair
Department of Computer Science

Mary Thomas
Department of Computer Science

Gustaaf Jacobs, Co-chair
Department of Aerospace Engineering

Dec 8, 2014
Approval Date
Copyright © 2015
by
Shivam Dixit
All Rights Reserved
DEDICATION

I would like to dedicate this thesis to my mother, father, Sister, Brother-in-law and friends who always supported, encouraged and stood by me for accomplishing my work. I also want to dedicate this thesis to my Professor Dr. Carl Eckberg, Dr. Gustaf Jacobs and Dr. Mary Thomas who guided me on every step throughout my Master’s.
ABSTRACT OF THE THESIS

Comprehensive Web Application for CITER
by
Shivam Dixit
Master of Science in Computer Science
San Diego State University, 2015

The emphasis of this thesis is to build an intuitive and robust Web Application for the CITER (Center of Industrial Training and Engineering Research) Department of San Diego State University.

The web application provides an overview of the department’s research work also includes the key members involved in the research that supports the CITER structure and enhances collaboration between industrial partners and SDSU’s Engineering departments, faculty members, undergraduate and graduate students with engineers and staff scientists of local industry. Through industry funded projects and scholarships, the students get first-hand training in an industry environment and are ready to join the workforce upon completion of their studies.

The web application is developed using HTML5, PHP to make a connection to SUN OS via Shell Scripts to execute MATLAB scripts hosted over the SUN OS Platform, and to generate images from data files for the horizontal and vertical velocity field near a temporally evolving boundary layer under surface the solitary wave.

The user interface, as well as the language, was intentionally kept simple and easy to use, to broaden the potential audience. To keep the user engaged in using the web application, key aspects are provided in the web application that eliminates the need of using a standalone MATLAB environment to generate the output image plots using HTML pages.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT ......................................................................................................................... v</td>
</tr>
<tr>
<td>LIST OF FIGURES .............................................................................................................. viii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS ....................................................................................................... x</td>
</tr>
<tr>
<td>CHAPTER</td>
</tr>
<tr>
<td>1 INTRODUCTION .............................................................................................................. 1</td>
</tr>
<tr>
<td>1.1 Overview ...................................................................................................................... 1</td>
</tr>
<tr>
<td>1.2 Motivation .................................................................................................................... 1</td>
</tr>
<tr>
<td>1.3 About the Application ................................................................................................. 3</td>
</tr>
<tr>
<td>2 TECHNOLOGIES .............................................................................................................. 4</td>
</tr>
<tr>
<td>2.1 PHP ............................................................................................................................. 4</td>
</tr>
<tr>
<td>2.2 History of PHP ............................................................................................................. 4</td>
</tr>
<tr>
<td>2.3 Shell Scripts and Sun OS ......................................................................................... 5</td>
</tr>
<tr>
<td>2.4 History of Sun OS or Solaris .................................................................................. 6</td>
</tr>
<tr>
<td>2.5 MATLAB Scripts and MATLAB ............................................................................. 6</td>
</tr>
<tr>
<td>2.6 History of MATLAB .................................................................................................. 6</td>
</tr>
<tr>
<td>2.7 HTML5, JavaScript, jQuery and CSS .................................................................. 8</td>
</tr>
<tr>
<td>2.7.1 HTML5 ................................................................................................................... 8</td>
</tr>
<tr>
<td>2.7.2 History of HTML5 ................................................................................................. 9</td>
</tr>
<tr>
<td>2.7.3 CSS (Cascading Style Sheet) .............................................................................. 9</td>
</tr>
<tr>
<td>2.7.4 History of CSS ...................................................................................................... 10</td>
</tr>
<tr>
<td>2.7.5 JavaScript ........................................................................................................... 10</td>
</tr>
<tr>
<td>2.7.6 History of JavaScript .......................................................................................... 11</td>
</tr>
<tr>
<td>2.7.7 jQuery ................................................................................................................ 11</td>
</tr>
<tr>
<td>2.7.8 History of jQuery ................................................................................................ 12</td>
</tr>
<tr>
<td>2.8 Software Prototype ................................................................................................. 12</td>
</tr>
<tr>
<td>2.9 Setting Up the Developer Environment ................................................................ 13</td>
</tr>
<tr>
<td>3 DESIGN AND ARCHITECTURE OF THE CITER WEBSITE .................................... 15</td>
</tr>
</tbody>
</table>
3.1 CITER Website Design ................................................................. 15
  3.1.1 Editing a Web Page .............................................................. 15
  3.1.2 The Page Content Area ....................................................... 15
  3.1.3 Editing a Free Text Module ................................................. 17
  3.1.4 Adding Titles to Free Text Modules ...................................... 18
  3.1.5 Adding and Editing Hyperlinks (Links) ............................... 18
  3.1.6 Click Insert/Edit Link Icon ................................................. 18
  3.1.7 Remove/Delete Link ........................................................... 19
  3.1.8 Uploading a Document ...................................................... 19
  3.1.9 Copy Document Link .......................................................... 20
  3.1.10 Adding Photos/Images ...................................................... 21
3.2 CITER Website Pages and Their Functions .................................. 21
  3.2.1 CITER Home Page ............................................................. 22
  3.2.2 About Us ........................................................................... 24
  3.2.3 People .............................................................................. 24
  3.2.4 Project ............................................................................. 24
    3.2.4.1 SynJet ....................................................................... 24
    3.2.4.2 STEM ....................................................................... 27
  3.2.5 Contact Us ......................................................................... 28
3.3 Running the CITER Website ......................................................... 30
4 CITERVIZ PROJECT ............................................................................. 31
  4.1 Implementation Details ............................................................. 31
  4.2 Relevant Code Snippets for CITERVIZ ..................................... 36
    4.2.1 Index.php ....................................................................... 36
    4.2.2 Final.php ....................................................................... 37
    4.2.3 Run.sh ........................................................................... 37
    4.2.4 LCS1.m ......................................................................... 37
    4.2.5 Index.php AJAX Script .................................................... 37
    4.2.6 Index.php jQuery Ready Stage ......................................... 38
5 FUTURE IMPROVEMENTS ................................................................. 39
6 CONCLUSION AND DEVELOPMENT CHALLENGES ...................... 40
BIBLIOGRAPHY ................................................................................. 42
APPENDICES

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>CITERVIZ PAGE INDEX.HTML</td>
</tr>
<tr>
<td>B</td>
<td>PHP SCRIPT FINAL.PHP</td>
</tr>
<tr>
<td>C</td>
<td>SHELL SCRIPT RUN.SH</td>
</tr>
<tr>
<td>D</td>
<td>MATLAB SCRIPT LCS1.M</td>
</tr>
<tr>
<td>E</td>
<td>JAVASCRIPT CITERUI.JS</td>
</tr>
<tr>
<td>F</td>
<td>CSS CITERCSS.CSS</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>CITER website’s home page.</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>CITERVIZ page.</td>
<td>3</td>
</tr>
<tr>
<td>3.1</td>
<td>Specifications on which the CITER website is created.</td>
<td>16</td>
</tr>
<tr>
<td>3.2</td>
<td>SDSU toolbox.</td>
<td>17</td>
</tr>
<tr>
<td>3.3</td>
<td>Editing a web page.</td>
<td>17</td>
</tr>
<tr>
<td>3.4</td>
<td>Adding page content area.</td>
<td>17</td>
</tr>
<tr>
<td>3.5</td>
<td>Pencil and free text icons.</td>
<td>17</td>
</tr>
<tr>
<td>3.6</td>
<td>Editing a free text module.</td>
<td>18</td>
</tr>
<tr>
<td>3.7</td>
<td>Adding titles to free text modules.</td>
<td>18</td>
</tr>
<tr>
<td>3.8</td>
<td>Adding and editing hyperlinks (links).</td>
<td>19</td>
</tr>
<tr>
<td>3.9</td>
<td>Uploading a document.</td>
<td>19</td>
</tr>
<tr>
<td>3.10</td>
<td>Downloadable document list module.</td>
<td>20</td>
</tr>
<tr>
<td>3.11</td>
<td>Downloadable document list module with options to create links.</td>
<td>20</td>
</tr>
<tr>
<td>3.12</td>
<td>Document link.</td>
<td>20</td>
</tr>
<tr>
<td>3.13</td>
<td>CITER website home page.</td>
<td>21</td>
</tr>
<tr>
<td>3.14</td>
<td>Custom CITER website logo.</td>
<td>22</td>
</tr>
<tr>
<td>3.15</td>
<td>Slider module added for rich user experience on the home page.</td>
<td>23</td>
</tr>
<tr>
<td>3.16</td>
<td>Custom navigation bar used across CITER website and CITERVIZ application.</td>
<td>23</td>
</tr>
<tr>
<td>3.17</td>
<td>Custom center text container.</td>
<td>23</td>
</tr>
<tr>
<td>3.18</td>
<td>Left Text Containers having image.</td>
<td>24</td>
</tr>
<tr>
<td>3.19</td>
<td>About us page of CITER website.</td>
<td>25</td>
</tr>
<tr>
<td>3.20</td>
<td>People page displaying all the profiles of faculty and students.</td>
<td>26</td>
</tr>
<tr>
<td>3.21</td>
<td>SynJet project page; defining SynJet and work done by CITER.</td>
<td>27</td>
</tr>
<tr>
<td>3.22</td>
<td>STEM project page; showcasing the students working on flow control.</td>
<td>28</td>
</tr>
<tr>
<td>3.23</td>
<td>CITER website’s contact us page.</td>
<td>29</td>
</tr>
<tr>
<td>3.24</td>
<td>Homepage for CITER.</td>
<td>30</td>
</tr>
<tr>
<td>4.1</td>
<td>Folder structure in EON.SDSU.EDU displayed by WinSCP.</td>
<td>32</td>
</tr>
</tbody>
</table>
Figure 4.2. Flowchart of CITERVIZ project. .................................................................32
Figure 4.3. CITERVIZ project page. .................................................................34
Figure 4.4. Index.php processing the files. .................................................................35
Figure 4.5. Computed result returned from MATLAB on the index.php........35
Figure 4.6. Help pop window; this is made for users to help creating the data files. ....36
ACKNOWLEDGEMENTS

I thank my advisor and thesis chairperson Professor Dr. Carl Eckberg, San Diego State University for giving me an opportunity to work on this thesis project and always supporting me with my work with sufficient guidance to accomplish this thesis.

I thank my thesis Co-Chair and Director of CITER Professor Dr. Gustaaf Jacobs, San Diego State University for his acceptance and being part of this Thesis committee.

I also am obliged to Professor Dr. Mary Thomas, San Diego State University for showing interest in knowing about the tool and its use and being part of this thesis committee.
CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

For any successful institution, gathering students and industries together is an important responsibility. With the advent of technology, any institution which works on research and engineering calls for a need to connect to the business worlds for providing engineering ideas and propositions in terms of projects to the industries in the relevant fields of engineering. At CITER we aim at working toward Mechanical and Aerospace Engineering projects, and therefore there is a need to showcase the work done by meticulous students to a larger audience and industries is required. This project is a prototype of the work that aims at generating output using numerous web technologies connecting to a server and computing results based on user input.

To showcase the world about the efforts these meticulous students put in, we therefore need a platform and the Comprehensive web application of CITER provides a major web platform to be presented over the web. Two key technologies were used to develop the CITER project:

- **CITER Web Module**
  The CITER Web Module is an overview of the CITER department about its work and complete information that you need to know about the CITER using a web interface (See Figure 1.1).

- **CITERVIZ Module**
  The CITERVIZ Module is also a web interface with focus on specific project done for the CITER department which aims as generating visualization images based on user input (See Figure 1.2).

1.2 MOTIVATION

The work of the people of CITER over the past few years is so vast that it is not possible to cover it all in this thesis so we decided to work on one of the key aspects of the CITER Project, which includes providing a comprehensive and interactive web application to
Figure 1.1. CITER website’s home page.

showcase the work of dedicated people involved in CITER. This thesis showcases the CITER website and all its custom built pages for its users. The thesis is also dedicated to CITERVIZ which features an interactive web module that utilizes the power of the world wide web; using a web application also provides a way of computing the results of MATLAB Based LCS applications on the web, based on user inputs in terms of data files; this would work for any web user who would like to compute values based on an LCS application. The project also provides a free solution to compute the results of an LCS Application, as the website is based on an Open source GNU license, and the code is executed over the SUNOS via Shell Script through server side code based on PHP, and connected through HTML5 with help from JavaScript, therefore mitigating the need for buying any licensed application such as MATLAB and SunOS.
1.3 ABOUT THE APPLICATION

CITER web application is a website based on collective information which provides details about the research work performed by the people at CITER. The Center of Industrial Training and Engineering Research is a department which combines traditional preparation of students with a new emphasis for industrial practice. This sets the ideal stage for an intensive collaboration between the College and a significant local industry such as Solar Turbines.

The CITER website contains detailed information about the people and their work in the field of Aerospace and Mechanical Engineering at San Diego State University and affiliated universities such as UCSD. The CITER website also contains a very intuitive and interactive web module, which aims at generating images based on data files. These files are taken as user input from a user local machine and then uploaded to a webserver hosted over a UNIX OS SUNOS using a PHP engine; the data files are then taken and evaluated in the server and then passed to a MATLAB script using a shell script which is hosted over a UNIX hosted MATLAB server.
CHAPTER 2

TECHNOLOGIES

This chapter explains the technologies and platforms used to develop this application. All technologies have their own advantages and disadvantages but the set of software technologies below were chosen to produce an effective product.

2.1 PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. PHP code can be simply mixed with HTML code, or it can be used in combination with various templating engines and web frameworks. PHP code is usually processed by a PHP interpreter, which is usually implemented as a web server's native module or a Common Gateway Interface (CGI) executable. After the PHP code is interpreted and executed, the web server sends resulting output to its client, usually in the form of a part of the generated web page – for example, PHP code can generate a web page's HTML code, an image, or some other data. PHP has also evolved to include a command-line interface (CLI) capability and can be used in standalone graphical applications.

What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server, generating HTML which is then sent to the client. The client would receive the results of running that script, but would not know what the underlying code was. You can even configure your web server to process all your HTML files with PHP, and then there's really no way that users can tell what you have up your sleeve.

2.2 HISTORY OF PHP

PHP development began in 1994 when Rasmus Lerdorf wrote a series of Common Gateway Interface (CGI) binaries in C which he used to maintain his personal homepage. He extended them to add the ability to work with web forms and to communicate with databases, and called this implementation "Personal Home Page/Forms Interpreter" or PHP/FI. [1, 2]
On May 22, 2000, PHP 4, powered by the Zend Engine 1.0, was released. As of August 2008 this branch reached version 4.4.9. PHP 4 is no longer under development nor will any security updates be released.

On July 13, 2004, PHP 5 was released, powered by the new Zend Engine II. PHP 5 included new features such as improved support for object-oriented programming, the PHP Data Objects (PDO) extension (which defines a lightweight and consistent interface for accessing databases), and numerous performance enhancements. In 2008 PHP 5 became the only stable version under development. Late static binding had been missing from PHP and was added in version 5.3.

Many high-profile open-source projects ceased to support PHP 4 in new code as of February 5, 2008, because of the GoPHP5 initiative, provided by a consortium of PHP developers promoting the transition from PHP 4 to PHP 5.

Over time, PHP interpreters became available on most existing 32-bit and 64-bit operating systems, either by building them from the PHP source code, or by using pre-built binaries. For the PHP versions 5.3 and 5.4, the only available Microsoft Windows binary distributions were 32-bit x86 builds, requiring Windows 32-bit compatibility mode while using Internet Information Services (IIS) on a 64-bit Windows platform. PHP version 5.5 made the 64-bit x86-64 builds available for Microsoft Windows.

2.3 SHELL SCRIPTS AND SUN OS

A shell script is a computer program designed to be run by the UNIX shell, a command line interpreter. The various dialects of shell scripts are considered to be scripting languages.

A shell script can provide a convenient variation of a system command where special environment settings, command options, or post-processing apply automatically, but in a way that allows the new script to still act as a fully normal UNIX command.

One example would be to create a version of ls, the command to list files, giving it a shorter command name of l, which would be normally saved in a user's bin directory as /home/username/bin/l, and a default set of command options pre-supplied.

SunOS is a version of the UNIX operating system developed by Sun Microsystems for their workstation and server computer systems. The SunOS name is usually only used to
refer to versions 1.0 to 4.1.4 of SunOS. These versions were based on BSD, while SunOS version 5.0 and later are based on UNIX System V Release 4, and are marketed under the brand name Solaris.

2.4 HISTORY OF SUN OS OR SOLARIS

In 1987, AT&T Corporation and Sun announced that they were collaborating on a project to merge the most popular UNIX flavors on the market at that time: BSD (including many of the features then unique to SunOS), System V, and Xenix. This would become System V Release 4 (SVR4).

On September 4, 1991, Sun announced that its next major OS release would switch from its BSD-derived source base to one based on SVR4. Although the internal designation of this release would be SunOS 5, from this point Sun began using the marketing name Solaris. The justification for this new "overbrand" was that it encompassed not only SunOS, but also the OpenWindows desktop environment and Open Network Computing (ONC) functionality.

Even though the new SVR4-based OS was not expected to ship in volume until the following year, Sun immediately began using the new Solaris name to refer to the currently shipping SunOS 4 release (also including OpenWindows). Thus SunOS 4.1.1 was rebranded Solaris 1.0; SunOS 5.0 would be considered a part of Solaris 2.0. SunOS 4.1.x micro versions continued to be released through 1994, and each of these was also given a Solaris 1.x equivalent name. In practice, these were often still referred to by customers and even Sun personnel by their SunOS release names. [3, 4]

2.5 MATLAB SCRIPTS AND MATLAB

MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment and fourth-generation programming language. Developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, Fortran and Python.

Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the MuPAD symbolic engine, allowing access to symbolic computing capabilities. An additional package, Simulink, adds graphical multi-domain simulation and Model-Based Design for dynamic and embedded systems.

2.6 HISTORY OF MATLAB

Cleve Moler, the chairman of the computer science department at the University of New Mexico, started developing MATLAB in the late 1970s. He designed it to
give his students access to LINPACK and EISPACK without them having to learn Fortran. It soon spread to other universities and found a strong audience within the applied mathematics community. Jack Little, an engineer, was exposed to it during a visit Moler made to Stanford University in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in C and founded MathWorks in 1984 to continue its development. These rewritten libraries were known as JACKPAC. In 2000, MATLAB was rewritten to use a newer set of libraries for matrix manipulation, LAPACK.

MATLAB was first adopted by researchers and practitioners in control engineering, Little's specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching of linear algebra and numerical analysis, and is popular amongst scientists involved in image processing. [5, 6]

The MATLAB application is built around the MATLAB language, and most use of MATLAB involves typing MATLAB code into the Command Window (as an interactive mathematical shell), or executing text files containing MATLAB code, including scripts and/or functions. [5]

There are 2 types of m-files:

- **Scripts**
- **Functions**

Scripts are a type of m-file that runs in the current workspace. So if you call a script from the command line (base workspace) the script will use and manipulate the variables of the base workspace. This can get very messy and lead to all sorts of strange errors when loops are involved and the coder is lazy about naming his loop variables (i.e. for i= 1:10, if every loop uses i, j, or k then it's likely that any script called from a loop will alter the loop variable).

Functions are wholly contained in themselves. They possess their own workspace keeping workspaces separate. This means that all variables necessary for a particular function must be passed or defined in some way. This can get tedious for complex algorithms requiring lots of variables. However, any manipulations of variables are discarded when the function is exited. Only those output arguments provided by the function are available to the calling workspace. This means that loops can use i, j, or k all they want because the function's workspace and the calling workspace do not mix.

Any command valid at the command line is valid in any m-file so long as the necessary variables are present in the m-files operating workspace.
Using functions properly any change can be affected to any algorithm or plotting tool. This allows for automation of repetitive tasks.

It is optional to end the M-file with 'end'; doing so, however, can lead to complications if you have conditionals or loops in your code, or if you're planning on using multiple functions in the same file (see nested functions for details on this).

### 2.7 HTML5, JavaScript, jQuery and CSS

HTML, Hypertext Markup Language is just a plain text file which is either saved as ‘.HTML.’ or ‘.htm’. This file contains small code snippets embedded inside different HTML tags, deciding how the content should be rendered to the user over a webpage. The HTML tags are placed inside angle brackets within which lies the content of the webpage and also these tags are placed in pairs marking the starting and the end point.

Any document can be structured with heading, anchors, lists, sub-lists, tables, paragraphs and other features using HTML. Another advantage of using HTML in any document would be the use of embedding video, audio and 2d graphic animations; web pages like these could be designed and put to good use using HTML.

Now that we know how a document can be rendered the next step would be to know how to make this document user friendly and interactive. This can be achieved using JavaScript which accesses all the contents on the rendered screen and dynamically updates and re-renders the screen contents. The JavaScript is usually written between tags `<script>…. </script>` and this can be included in the HTML file or can be written as a separate file with ‘.js’ extension.

The most important feature of a webpage is presentation; this is taken care by CSS (Cascading Style Sheets). Like JavaScript this can also be part of the main page alongside HTML tags defining how it should be rendered (color, font, size etc.), or can be written into a separate ‘.CSS’ file which helps for better readability and brevity.

#### 2.7.1 HTML5

HTML5 is a core technology markup language of the Internet used for structuring and presenting content for the World Wide Web. As of October 2014 this is the final and complete fifth revision of the HTML standard of the World Wide Web Consortium (W3C). The previous version, HTML 4, was standardized in 1997.
Its core aims have been to improve the language with support for the latest multimedia while keeping it easily readable by humans and consistently understood by computers and devices (web browsers, parsers, etc.). HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML.

Following its immediate predecessors HTML 4.01 and XHTML 1.1, HTML5 is a response to the fact that the HTML and XHTML in common use on the World Wide Web are a mixture of features introduced by various specifications, along with those introduced by software products such as web browsers, those established by common practice, and the many syntax errors in existing web documents. It is also an attempt to define a single markup language that can be written in either HTML or XHTML syntax. It includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a potential candidate for cross-platform mobile applications. Many features of HTML5 have been built with the consideration of being able to run on low-powered devices such as smartphones and tablets.

2.7.2 History of HTML5

The Web Hypertext Application Technology Working Group (WHATWG) began work on the new standard in 2004. At that time, HTML 4.01 had not been updated since 2000, and the World Wide Web Consortium (W3C) was focusing future developments on XHTML 2.0. In 2009, the W3C allowed the XHTML 2.0 Working Group's charter to expire and decided not to renew it. W3C and WHATWG are currently working together on the development of HTML5.

While HTML5 is often compared to Adobe Flash, the two technologies are very different. Both include features for playing audio and video within web pages, and for using Scalable Vector Graphics. HTML5 on its own cannot be used for animation and interactivity — it must be supplemented with CSS3 or JavaScript. There are many Flash capabilities that have no direct counterpart in HTML5. See any comparison of HTML5 and Flash. [7, 8]

2.7.3 CSS (Cascading Style Sheet)

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. While most often used to change the style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. Along with
HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .CSS file, and reduce complexity and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers. CSS makes it possible to separate presentation instructions from the HTML content in a separate file or style section of the HTML file. For each matching HTML element, it provides a list of formatting instructions.

2.7.4 History of CSS

CSS was first proposed by Håkon Wium Lie on October 10, 1994. At the time, Lie was working with Tim Berners-Lee at CERN. Several other style sheet languages for the web were proposed around the same time, and discussions on public mailing lists and inside W3C resulted in the first W3C CSS Recommendation (CSS1) being released in 1996. In particular, Bert Bos' proposal was influential; he became co-author of CSS1 and is regarded as co-creator of CSS.

Style sheets have existed in one form or another since the beginnings of Standard Generalized Markup Language (SGML) in the 1980s, and CSS was developed to provide style sheets for the web. One requirement for a web style sheet language was for style sheets to come from different sources on the web. Therefore, existing style sheet languages like DSSSL and FOSI were not suitable. CSS, on the other hand, allowed a document's style to be influenced by multiple style sheets by way of "cascading". [9, 10]

2.7.5 JavaScript

JavaScript is a dynamic computer programming language. It is most commonly used as part of web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed. It is also used in server-side network programming with frameworks such as Node.js, game development and the creation of desktop and mobile applications.
JavaScript is classified as a prototype-based scripting language with dynamic typing and first-class functions. This mix of features makes it a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles.

Despite some naming, syntactic, and standard library similarities, JavaScript and Java are otherwise unrelated and have very different semantics. The syntax of JavaScript is actually derived from C, while the semantics and design are influenced by the self and Scheme programming languages.

JavaScript is also used in environments that aren't web-based, such as PDF documents, site-specific browsers, and desktop widgets. Newer and faster JavaScript virtual machines (VMs) and platforms built upon them have also increased the popularity of JavaScript for server-side web applications. On the client side, JavaScript has been traditionally implemented as an interpreted language, but more recent browsers perform just-in-time compilation.

JavaScript has been standardized in the ECMA Script language specification [11].

### 2.7.6 History of JavaScript

JavaScript was originally developed by Brendan Eich, while working for Netscape Communications Corporation. While competing with Microsoft for user adoption of web technologies and platforms, Netscape considered their client-server offering a distributed OS with a portable version of Sun Microsystems' Java, providing an environment in which applets could be run.[citation needed] Because Java was a competitor of C++ and aimed at professional programmers, Netscape also wanted a lightweight interpreted language that would complement Java by appealing to nonprofessional programmers, like Microsoft's Visual Basic (see JavaScript and Java).

Although it was developed under the name Mocha, the language was officially called LiveScript when it first shipped in beta releases of Netscape Navigator 2.0 in September 1995, but it was renamed JavaScript when it was deployed in the Netscape browser version 2.0B3. [12]

### 2.7.7 jQuery

jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. Used by over 60% of the 10,000 most visited websites, jQuery is the most popular JavaScript library in use today. jQuery is free, open source software, licensed under the MIT License [13].
jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, theme-able widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and web applications.

The set of jQuery core features—DOM element selections, traversal and manipulation—enabled by its selector engine (named "Sizzle" from v1.3), created a new "programming style", fusing algorithms and DOM data structures. This style influenced the architecture of other JavaScript frameworks like YUI v3 and Dojo, later stimulating the creation of the standard Selectors API.

2.7.8 History of jQuery
jQuery was originally released in January 2006 at BarCamp NYC by John Resig and was influenced by Dean Edwards' earlier cssQuery library. It is currently maintained by a team of developers led by Dave Methvin. [14]

2.8 SOFTWARE PROTOTYPE
Software prototyping is an important component for the development of any software. It is the activity of creating incomplete versions of product under development. It drives the whole process and warns about the challenges coming ahead. It serves as an initial model which is continuously refined as the development proceeds towards meeting the requirement [15].

There is a stepwise approach to designing a software prototype:

- Requirement gathering and identification – This step involves the basic requirements of the product, like the interface and outlook.
- Developing initial prototype – The prototype developed in this stage showcases the basic requirement and interface. The features may or may not be implemented or work in the same manner as the software is supposed to be.
- Review – The prototype is then presented to the customer or interested party and their feedback are collected to know the direction of future work for the application.
- Revise and Enhance- The feedback is worked upon and then brought into action and the product is enhanced to develop the new prototype and this cycle repeats till the expectations are met.
The initial prototype for this application comprised of webpages which showed the representation of the work done by the CITER students and faculty. This was later enhanced to incorporate user interaction by providing the CITERVIZ module which included visualization based on representation of MATLAB outputs in the form of images on to the web module.

At every stage there was more scope to improve, revise and to provide better knowledge to the user. Hence this application is refined to develop a final and effective web tool which can provide the user with all information they need to compute over MATLAB in a much easier way over the internet using their browser.

### 2.9 Setting Up the Developer Environment

We have already discussed about the various technologies that were used for this project, and we would therefore focus here more about the development environment and how to setup that along with some prerequisites that were required during the development of this thesis project.

**SYSTEM Perquisites:**
1. Web Browser
2. OpenSSH/Putty
3. Firezilla/WINSCP

**Development Environment:**
2. HTML Web Editor (codeanywhere.com)[16]
3. UNIX Environment (Ubuntu 14.0.10.1)

**Production Environment:**
1. CMS Toolbox for SDSU
2. Linux hosted Webserver
3. PHP hosted on Web Server
4. MATLAB hosted over Linux.

**Setting up the web development Environment:**
1. Open Visual Studio
2. Select New Web Project
3. Add Existing Item to Project. Select HTML Page.
4. Add the code to work on by including custom reference to jQuery, CSS and JavaScripts.

5. Using Firezilla/WinSCP to move files from Local Machine on to Web Server.

Setting up the server side development Environment:

1. Connect to UNIX server using Putty/OpenSSH
2. Open Terminal
3. Install PHP
4. Run Test PHP Script from Terminal
5. Run Test Shell Script from Terminal
6. Run Test MATLAB Script from Terminal
CHAPTER 3

DESIGN AND ARCHITECTURE OF THE CITER WEBSITE

The CITER website is an easy to use website; with its easy and interactive modules or pages anybody can get information that they need to know about CITER. The CITER website is also made in compliance with the SDSU standard to be in coherence with accessibility with partially visually disabled users. The website link and images have anchor text which also allows images to be read by special software for the visually challenged. This topic is out of the scope of this thesis.

3.1 CITER WEBSITE DESIGN

The CITER website is made using the CMS Toolbox provided by the San Diego State University, center of Media and Information Exchange and using their layout format specified as in Figure 3.1.

For making any website for using sdsu.edu domain the website/application must be created using the newscenter.sdsu.edu/toolbox

The newscenter is a CMS tool provided by SDSU, Media and Information Exchange, which provides the tools for creating interactive websites (See Figure 3.2).

Following are some of the key features involved in Creating/Editing websites on Toolbox.

3.1.1 Editing a Web Page

1. Click on the "+" icon to expand the category of the page you want.

2. Click on the name of the page you would like to edit (this will take you to the Edit Page). (See Figure 3.3.)

3.1.2 The Page Content Area

At the bottom of the Edit Page is the PAGE CONTENT area where you can edit free text modules (See Figure 3.4).
Figure 3.1. Specifications on which the CITER website is created.
3.1.3 Editing a Free Text Module

To edit a module, click on the pencil icon. Note: Clicking on the trash can icon will delete the module (See Figure 3.5). You will get a prompt box to confirm your selection. At this time there is no way to restore a deleted module. Always keep a backup copy of your web page text.

You will be brought to the Content Editor where you can edit the page much like you would in a Microsoft Word document.
It makes use of the Format Styles to keep content styled consistently throughout.

Paragraph: Body text should be formatted with this style (See Figure 3.6).

Figure 3.6. Editing a free text module.

3.1.4 Adding Titles to Free Text Modules

In the Free Text module page, there is a second tab next to the Content editor tab called “Settings.” Go to this tab and give your module a title. As your page has more and more text modules it will be easier to get to the module you want if it has a short descriptive title (See Figure 3.7).

Figure 3.7. Adding titles to free text modules.

3.1.5 Adding and Editing Hyperlinks (Links)

Highlight the text you would like linked.

3.1.6 Click Insert/Edit Link Icon

In the Insert/edit Link window enter a Link URL (this could be a link to a website or to a document) and a title for the link. Leave the Link Target as default "Open in This Window" (target="_self"). (See Figure 3.8).
Click Insert. You will be taken back to the Content Editor.

Click Submit to save your changes.

Use Descriptive Link Text.

### 3.1.7 Remove/Delete Link

- Click anywhere on your linked text.
- Click the Unlink icon to break the link.
- Click Submit to save your changes.

### 3.1.8 Uploading a Document

- Create a document and save as a PDF file (unless there's a particular reason you need an editable file, stick to PDF).
- Click the “+” icon to expand the Documents category in your department or program.
- Click on the documents page (See Figure 3.9).

A special kind of module is used to upload documents. It is called the Downloadable Document List module.

Find the “Downloadable Document List” module in the PAGE CONTENT area. (See Figure 3.10)
Figure 3.10. Downloadable document list module.

- Click on the pencil icon.
- In the document module form, type a document title (that will help you find that document later in the document list).
- Click Choose File to attach your document.
- Click Submit to add your document to documents list (located below the form). (See Figure 3.11).

![Figure 3.11. Downloadable document list module with options to create links.](image)

3.1.9 Copy Document Link

- Find your document in the document list.
- Right click (or Ctrl + click for single-button mouse) on the magnifying glass icon.
- Choose Copy Link Address to copy the location of your document (note: the option may read Copy Link Location or Copy Link depending on your browser). (See Figure 3.12).

![Figure 3.12. Document link.](image)
3.1.10 Adding Photos/Images

Tips:

- Make sure photos are good quality images. If they are not good quality, don’t use them.
- Images should be saved at 72PPI (Pixels Per Inch).
- Images should be cropped to the size needed to fit into the space on the page. Note: left column width is 550 pixels; right column width is 285 pixels.
- Give your image a descriptive file name to make it easy to find later in the site’s stored images.
- Only use underscores ( _ ), letters and numbers in filenames.

3.2 CITER WEBSITE PAGES AND THEIR FUNCTIONS

As already mentioned CITER is a user friendly web application; in this section we are going to discuss the various pages in detail about their purpose and the underlying technology on which they are built (See Figure 3.13).

Figure 3.13. CITER website home page.
3.2.1 CITER Home Page

The homepage for CITER is comprised of various web parts comprising of:

- Custom Header Image.

  The Header Image is made in paint with embossing over a image of jet stream of air. The color of the Title on the header image is maintained to comply with SDSU design methodology (See Figure 3.14).

![Custom CITER Website Logo](image)

**Figure 3.14. Custom CITER website logo.**

This header image is consistent throughout the website and each page has this image as their header Image.

- A module called Slider for rotation of Image

  The above module is called as a slider module; this can take up to 12 images/videos for display. These 12 images are further divided into two different parts the Larger Images and the smaller icon images for each larger Image (See Figure 3.15).

- Menu Navigation bar

  CITER navigation is the main command central for the CITER website and the CITERVIZ Web Application. The navigation is made using the HTML5 by using the ui and the li tags (See Figure 3.16). These tags are then optimized by the CSS script to for a flattened user experience and links are provided using JavaScript.

  Like the header this is also being used in the entire CITER website and the CITERVIZ Web page.

- Custom Center Text Container

  The above container is a custom container generated by combining HTML divisions and combining into one. The divisions are merged by using the CSS (See Figure 3.17).

- Two Custom Left Text Containers
Figure 3.15. Slider module added for rich user experience on the home page.

Figure 3.16. Custom navigation bar used across CITER website and CITERVIZ application.

Figure 3.17. Custom center text container.

The Center for Industrial Training and Engineering Research (CITER), structures and enhances collaboration between industrial partners and SDSU.

The primary objective is to connect SDSU’s Engineering departments, faculty members, undergraduate and graduate students with engineers and staff scientist of local industry.

Through industry funded projects and scholarships, the students get first-hand training in an industry environment and are ready to join the workforce upon completion of their studies.

CITER is eager to work with enthusiastic students and engineering companies in the San Diego area.

- Dr. Gusaar Jacobs
  Director CITR
Figure 3.18 shows the two text containers which can have images and text as free HTML input. These containers are based on HTML5.

![Image](image.jpg)

**Figure 3.18. Left Text Containers having image.**

### 3.2.2 About Us

The about us page is a custom built page with features such as custom navigation menu and custom header and footers.

The About Us page showcases the director Message and also includes a short description about Dr. Gustaaf Jacobs, DIRECTOR of CITER; and includes the mail address to contact him (See Figure 3.19).

### 3.2.3 People

The People page showcases the key members and the scientists at CITER. Each member has details about them along with a link to their personal profiles (See Figure 3.20).

The page is created using the ToolBox CMS incorporating the custom header and Footer. The page also includes the custom navigation menu.

### 3.2.4 Project

The Project page gives it user to look into the detailed information about some of the key projects being done at CITER. The Project page highlights the key projects done at CITER; SynJet and STEM. I would like to explain them in following sub chapter.

#### 3.2.4.1 SYNJET

SynJet is research work done by the CITER students in which we focus at the synthetic jet control method to enhance the vane velocity (See Figure 3.21).
Figure 3.19. About us page of CITER website.
Figure 3.20. People page displaying all the profiles of faculty and students.
Figure 3.21. SynJet project page; defining SynJet and work done by CITER.

This page includes a video embedded into the age which auto starts up on page load. The Page complies with initial theme comprising of same custom header and footer along with the custom navigation menu.

### 3.2.4.2 STEM

STEM page focusses on the ongoing collaborative initiative between the San Diego MESA (Mathematic, Engineering, and Science Achievement) Alliance (SDMA). STEM is focused on research on mixing and flow control in High Speed Combustors (See Figure 3.22).
3.2.5 Contact Us

The Contact Us Page is created using the help of ToolBox that contains the CITER website theme. This page provides detailed information about the CITER department along with links that guide users to the location and the printable maps created in PDF for users to download.
This page also contains a Google maps link which will provide the distance and location from the user’s location. This is accomplished by adding Google Maps API (See Figure 3.23).

![CITER website's contact us page.](image)

**Figure 3.23. CITER website's contact us page.**

This page along with all other pages of CITER website and CITERVIZ application contain a SDSU search bar. This search bar is propriety search bar which is incorporated in this website and the CITERVIZ application to allow the users to search both across the website and the SDSU.
3.3 Running the CITER Website

Use any Browser to connect to CITER.sdsu.edu; the website is hosted over SSL and would therefore redirect you to https://CITER.sdsu.edu. CITER is hosted over SDSU propriety center of internet and media department https://newscenter.sdsu.edu

Hypertext Transfer Protocol Secure (HTTPS) is a communications protocol for secure communication over a computer network, with especially wide deployment on the Internet. Technically, it is not a protocol in and of itself; rather, it is the result of simply layering the Hypertext Transfer Protocol (HTTP) on top of the SSL/TLS protocol, thus adding the security capabilities of SSL/TLS to standard HTTP communications. The main motivation for HTTPS is to prevent wiretapping and man-in-the-middle attacks.

With its easy to use the navigation bar any user can easily access each page of the web site and can also look into the interactive modules using a single click from their mouse (See Figure 3.24).

![Figure 3.24. Homepage for CITER.](image)

The purpose and usage of pages of the CITER web site and links are explained in the next chapters.
CHAPTER 4

CITERVIZ PROJECT

CITERVIZ is one of the most user interactive modules of this thesis which allows the users to compute their data files for the MATLAB LCS Application. This is a one of a kind advancement done by the CITER people.

4.1 IMPLEMENTATION DETAILS

CITERVIZ is a project in which multiple technologies are combined to execute MATLAB code from an HTML web page hosted over the webserver. The CITERVIZ Project is hosted on the eon.sdsu.edu network, where all the files are created from scratch to incorporate the entire web site theme. To accomplish this, cascading style sheet (CSS) is made to have the same user experience as the CITER.sdsu.edu. Also using JavaScript, the menu navigation is controlled and structured to perform the same functionality as CITER.sdsu.edu.

As shown in Figure 4.1 all the files are stored in the web server in the public_HTML folder. The Index.php (See Appendix A) is the CITERVIZ Page in which the entire client side validations are done using jQuery and JavaScript and the Server side connections and validations are done using the file final.php (See Appendix B) which calls the Run.sh (See Appendix C) shell script to execute the MATLAB code (See Appendix D) on the UNIX terminal and generates an image based on clients local IP and local machine time.

The folder name is the most important feature of this visualization web application. The entire user provided data is stored in the form of text and ASCII .dat files in user specific directory. When a user uploads files at first the directory is created with a name that is generated by the IP address of the user machine. Upon this all the files are save in this directory.

MatabSourceCode folder contains all the MATLAB scripts which are required for the CITERVIZ to run the MATLAB code. The Figure 4.2 describe flowchart of generating the images based on user input.
CITERVIZ Project is a web interactive module by which any user can upload files in the form of data in extensions such as “.txt” and “.dat”. These files are restricted by the JavaScript (See Appendix E) and jQuery Code to allow only these file mime types to be uploaded by the user, hence checking the security concern.

These data files are then sent to the web server for evaluation and the data files are evaluated based upon their naming convention and the file sizes for the computation. The PHP script is responsible for evaluation and uploading the files to the web server. The PHP script also modifies the name for the MATLAB code execution as the MATLAB code need a special kind of naming system. The naming convention used for computing results by this
MATLAB is “velbn_<Number Format>.dat”. The PHP code also set the number of the uploaded files in a sequence of numbers starting from 0 as velbn_0.dat is the host MATLAB data file for the execution of MATLAB Binary File.

Shell Script is a middleware connector language used to connect both PHP and MATLAB. These two engines are hosted over the UNIX system called Solaris or SunOS. The PHP Engine uploads all selected files from the user’s local machine and encapsulates the files and makes the files ready for the MATLAB engine by passing these files via the Shell Script to the MATLAB Engine. The MATLAB engine computes MATLAB code written in the LCS.m script using the data and generates the images which are hence posted over the CITERVIZ page.

Since the MATLAB is hosted over the UNIX system we cannot display the image/contour output from the shell terminal and therefore the MATLAB code has been optimized to convert every contour image into figures, which are then saved as JPEG image Files. The MATLAB code is modified to convert every contour image to a JPEG image as JPEGs are very light in size and can be easily displayed on the web.

The user has the flexibility to upload as many data files he/she wants to send over the webservice for computation. To mitigate the issue of disk full error, we have incorporated an intuitive way to delete all the files after the entire execution process is finished. We could have done this by a timer job, or a server script to delete the user files, but by including this way we may encounter a deadlock as there could be simultaneous users running the MATLAB Engine to execute their data file computation. To avoid such situations we incorporated a better way to delete all the data files after the execution of the entire code. The data files will be deleted only when the code has been executed and the images are generated. This would therefore never lead to the server going to a disk full stage. This idea is incorporated in the PHP script which itself deletes the files after the execution of the entire MATLAB Code.

As mentioned earlier, the MATLAB computed results are stored in the form of images which are then picked up by the web page to be displayed on the CITERVIZ page. This is basically done using the technology called jQuery which basically picks up the MATLAB generated image and displays it on to the CITERVIZ page. JQuery on CITERVIZ page is responsible for the entire showing and removing of the images onto the web.
Whenever the MATLAB Code generates an image, the default image on the CITERVIZ page is replaced by this newly generated image. Also by using jQuery we are able to refresh the image every time a page has been refreshed.

The code snippet for this page is written in Appendix (See Appendix A through F) CITERVIZ Project page, this showcases the visualization based upon inputs provided by the users. This page is based on PHP and is hosted over eon.sdsu.edu (See Figure 4.3).

![CITERVIZ project page](image)

*Figure 4.3. CITERVIZ project page.*

As already mentioned about this project in previous chapters, we would like to mention some of the key features of this page in detail here.

This page is created in PHP, which allows user to submit data and then computes the user data based in files over MATLAB and shows the desired image back on the page. The page includes AJAX call to the server in which the user data files are sent over to the server using HTTP post request. The AJAX also provides the showing of processing image on the action of PHP upload (See Figure 4.4)

This page; like all other pages also includes the same custom header and footer. The custom navigation is again created in the page to incorporate the CITER website theme. CiterMenus.CSS (See Appendix F) is used mainly for the similar user experience for creating this webpage and custom JavaScript’s are used to replicate the same behavior of the other CITER website.
On this page when a HTTP request is sent to the server with the user data files a new directory is created with the users IP address and all the data files are loaded into the folder. The MATLAB script then uses these data files and computed results is presented in the form of image; this image is also stored in the same user directory. The CITERVIZ then takes the HTTP response message in the form of Header and display the specific user image on to the web page (See Figure 4.5).
This page also contains a small informative pop-up window which is used to helping the users to create their data files.

This pop-up window is a HTML page which has link to demo data files, the users can take demo data files from here and create their own files based on the structure provided in these files or can use these files to these the output. The data files are in text format and that can be opened in any browser (See Figure 4.6).

![Help pop window](image)

**Figure 4.6. Help pop window; this is made for users to help creating the data files.**

4.2 Relevant Code Snippets for CITERVIZ

As mentioned above, CITERVIZ is a visualization web application. This web application is based on various technologies such as JavaScript, AJAX and PHP, that computes the shell script and by passing directory variables such as data and directory information to compute the MATLAB code and generate a user specific image.

In this section we would like to discuss some of the important code from different technology and languages that were used in the execution of the CITERVIZ code and web application. Appendix A through F contains all the code that is used for creation of CITERVIZ and CITER web site.

4.2.1 Index.php

```php
<?php
    if(isset($_GET["img"])) {
        $n = $_GET["img"];
        $finalname = "finalname";
        echo $finalname;
        echo "<img src="$finalname/$n">";
    }
```
The above code snippets set the image that is created for a specific user after the HTTP response is sent from the final.php page.

### 4.2.2 Final.php

```php
$cmd = "./Run.sh $finalname &2>1";
shell_exec($cmd);
```

The above command takes filename that execute the shell command by passing a variable, this variable is the directory name that are sent over to the MATLAB code for looking into the directory for user submitted files.

### 4.2.3 Run.sh

```bash
#!/bin/sh
cd MatlabSourceCode
matlab -nodesktop -nojvm -nosplash -r "LCS1('$1') ; quit"
```

Since, the entire code is present in the MatlabSourceCode directory, the shell command first change the present working directory and then execute the MATLAB command with passing the user directory information.

### 4.2.4 LCS1.m

```matlab
Figure3 = figure('Visible','off');
contourf(Xpi{1}(2:2:gSize(1)-1,2:gSize(2)-1),Xpi{2}(2:gSize(1)-1,2:gSize(2)-1)...
,'FTLE,80,'LineStyle','None')
daspect([1 1 1])
title('This is your Output')
cd (strcat('..', path1));
saveas(Figure3,[path1 '.jpg']);
clear all;
```

By the above code the, first a figure handle is created and then then the contour is created. Since we are running MATLAB in UNIX system we cannot see the visual image so the image is save into the user directory folder using path1 variable.

### 4.2.5 Index.php AJAX Script

```javascript
var ray={
ajax:function(st)
{
this.show('load');
}
},
show:function(el)
{
```
this.getID(el).style.display="";
},
getID:function(el)
{
    return document.getElementById(el);
}
}

The code above shows a simple ajax call in which the a show message is set to true. This is used in showing the processing image that comes up when the file are being uploaded.

### 4.2.6 Index.php jQuery Ready Stage

$(document).ready(
    function(){
        $('input:file').change(
            function(){
                if ($this.val()) {
                    $('input:submit').attr('disabled',false);
                }
            }
        );
    });

The code above looks for the files are being uploaded and set the visibility of the upload button to disabled.
CHAPTER 5

FUTURE IMPROVEMENTS

Following the advent of the CITER website and the CITERVIZ application, there are many ideas which can be incorporated into the web application; we would like to present a few ideas that we look forward to being included.

The CITERVIZ application is dependent on the data files submitted by the user, which forces the user to make the data files correct and this may result in errors if the data is not correct. We would suggest that there could be features to allow users to generate data files on the fly, through the web page.

The CITERVIZ application is set up to run only one MATLAB script; the code is written to include multiple MATLAB scripts to be run. There could be a way to select different MATLAB files and include the entire server side script to work for different scripts.

The CITERVIZ application includes a deletion script that deletes the folder giving created by the different users. The deletion script works only for the administrator and therefore can be optimized to delete based on administrator request.
CHAPTER 6

CONCLUSION AND DEVELOPMENT

CHALLENGES

The CITER website and CITERVIZ are both catered to fulfill the need of its users. The CITER website is up and running from few months. The first phase of CITERVIZ was launched in December 2014.

There were some major challenges that were fought and removed from our way to release both the CITER web site and CITERVIZ application. Here are few of those:

Inability to use and scripting languages in ToolBox CMS was one of the major setbacks for the CITER such as client side ex. JavaScripts and server side ex. PHP. Since ToolBox is mandatory for creating SDSU domain websites, all the pages are created as an IFramed HTML pages included in ASPX pages.

To overcome this; custom calling pages were created for the CITER web site which could host or link JavaScript’s on them to be called and redirected for the navigation menu. However, by implementing this approach also we could not accomplish the complex current site pages such as CITERVIZ.

For creating CITERVIZ project which was based on heavy client side validation and major server side processing the above approach was not possible. So we decided to move to new domain eon.sdsu.edu, where we were allowed to create custom web pages.

There was a major setback in connecting to UNIX system from web, which is sending the HTTP post request to the server, many technologies were tried to implement such as Node.js and Ajax Term for UNIX, but these technologies required to be installed on the UNIX server. So to avoid all the hassles of new Implementation of technology, a way to connect both the web and the server, PHP was used. PHP was first installed and then the entire server side scripting was introduced to the CITERVIZ project.

Lastly for the CITERVIZ project we wanted to allow multiple users to be performing multiple server requests on the MATLAB processor. Therefore to achieve this numerous suggestions were given and took time to implement those. So we come up with a resolution
of creating user specific folders and uploading there specific data and the computed MATLAB image in them. These folders are created on a combination of IP address and time instance.

The image that is returned to the user should be user specific, so this was accomplished by making few changes in the MATLAB to generate the Images with name of same combination of IP address and time.
BIBLIOGRAPHY

WORKS CITED


WORKS CONSULTED


THE PHP GROUP. *Homepage*. The PHP Group, http://php.net/

P. M. REINHEIMER AND C. SHIFLETT, *Professional Web APIs with PHP*, Indianapolis, Ind, Wrox, 2006, ch. 4, 6, 9, 10, 12.

STACKOVERFLOW. *Homepage*. Stackoverflow, https://stackoverflow.com/

APPENDIX A

CITERVIZ PAGE INDEX.HTML
San Diego State University is one of the nation's premier urban research universities actively engaged in preparing graduates who will make an impact on the San Diego region and beyond.
$(document).ready(function() {
    var searchText = "search";

    var textBox = $('input[name="dpgMyPage$ctl11$txtSearch"]
    textBox.attr('value', searchText);
    textBox.click(function() {
        var txt = $('input[name="dpgMyPage$ctl11$txtSearch"]
        if (txt.attr('value') == searchText) {
            txt.attr('value', '')
        }
    });
    textBox.blur(function() {
        var txt = $('input[name="dpgMyPage$ctl11$txtSearch"]
        if (txt.attr('value') == '') {
            txt.attr('value', searchText)
        }
    });
});

<div class="search" id="dpgMyPage_ctl11_pnlSearch" onkeypress="javascript:return WebForm_FireDefaultButton(event, 'dpgMyPage_ctl11_btnSearch')"/>
<input type="text" class="search_textbox" id="dpgMyPage_ctl11_txtSearch" name="dpgMyPage$ctl11$txtSearch" title="Search Box" />
<input alt="Submit Search" class="search_button" id="dpgMyPage_ctl11_btnSearch" name="dpgMyPage$ctl11$btnSearch" src="https://newscenter.sdsu.edu/include/images/home/search_btn.png" style="height:22px;width:29px;border-width:0px;" title="Search Box" type="image" />
</div>
</div>
CiterViz is an interactive data visualization module, by this module we present you the MATLAB generated graph upon the data you upload. The MATLAB code will generate graph based on the velocity values provided in the data files you have uploaded. Your data files are safe and we do not keep your data in our server as your data files are automatically deleted after execution.

Please upload the data files using the right section of this page.

You can also read the help file provided in the end of the right section for how to create data files and time files. Also you can download sample files for each type of files.

<?php
//include 'final.php';
if(isset($_GET['img'])){
    $n=$_GET['img'];
    <table id="load" align="center" style="display:none;">
        <tr>
            <td contenteditable="true">
                <table style="text-align:center;">
                    <tr>
                        <td><img src="Processing.gif" alt="Processing" width="500" height="300"></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
};
echo $finalname;

echo "<img src='$finalname/$n'";  

?>

</div>
</td>
</tr>

<tr>
<td><br></td>
</tr>
</table>
</div>
<br>
</div>
<div style="clear:both"></div>
</div>
<div class="content_inner_left_bottom"></div>
</div>
<div class="content_right" id="dpgMyPage_pnlTemplateContentRight">
<div class="content_inner_right_top"></div>
<div class="content_inner_right_middle">
<a id="16583" name="16583"></a>
<div class="module free_text_module" id="dpgMyPage_ctl15_pnlFreeTextModule">
<table>
<tr><td><h2>Upload Binary Data</h2></td>
</tr>
<tr>
<td><div class="demo-conversion">
Upload Data and Timed files using after selecting files from your local machine the <b>Upload Button</b>.</td>
</tr>
</table>
</div>
</div>
</div>
</div>
<br>
</div>

<b>Note:</b> <i>Use files for different time instances.</i>
<br/>
<br/>

</div>

<div id="promoNode">
<script type="text/javascript">

</script>
var ray={
ajax:function(st)
{
    this.show('load');
    //this.show('newimage');
},
show:function(el)
{
    this.getID(el).style.display='';
},
getID:function(el)
{
    return document.getElementById(el);
}
}
</script>
<form action="final.php" method="post" enctype="multipart/form-data" onsubmit="return ray.ajax()">
<input type="file" id="file" name="files[]" onchange="makeFileList();" multiple="multiple"
accept=".txt, .dat" />
</p>
<br/>
<br/>
<p><strong>Files You Selected:</strong></p>
<ul id="fileList">
<li>No Files Selected</li>
</ul>
<br/>
<script>
$(document).ready(function(){
    $("input:file").change(function(){
        if ($(this).val()) {
            $("input:submit").attr("disabled",false);
        }
    });
});
</script>
<input type="submit" id="submit" value="Upload" disabled />
<script type="text/javascript">
function makeFileList() {
    var input = document.getElementById("file");
    var ul = document.getElementById("fileList");
    while (ul.hasChildNodes()) {
        ul.removeChild(ul.firstChild);
    }
    for (var i = 0; i < input.files.length; i++) {
        ul.appendChild(input.files[i]);
    }
</script>
```javascript
var li = document.createElement("li");
li.innerHTML = input.files[i].name;
ul.appendChild(li);
}
if(!ul.hasChildNodes()) {
    var li = document.createElement("li");
    li.innerHTML = 'No Files Selected';
    ul.appendChild(li);
}
```

<br><br>
**Please read the Help file from the below link how to create relevent files for the upload.**

```
<a href="Help.html" onclick="javascript:void(window.open('Help.html','1417583507531','width=700,height=300,toolbar=0,menubar=0,location=0,status=1,scrollbars=1,resizable=1,left=0,top=0');return false;”)">Link</a>
```

```
<div id="dpgMyPage_ctl15_pnlFTMContent"></div>
</div>

<div style="clear:both"></div>

<div class="content_inner_right_bottom"></div>
</div>

</div>

```
-- Content For Disqus --
```
APPENDIX B

PHP SCRIPT FINAL.php
<?php

header('Location: '.$_SERVER['HTTP_REFERER']);
$ip = $_SERVER['REMOTE_ADDR']?:($_SERVER['HTTP_X_FORWARDED_FOR']?:$_SERVER['HTTP_CLIENT_IP']);
date_default_timezone_set("UTC");
$date = date("Y-m-d-H-i-s", time());
$finalname=$ip.$date;
$finalname= str_replace(""," ",$finalname);
mkdir($finalname);

$i = 0;
foreach ($_FILES['files']['name'] as $f => $name) {
    $gridfile=pathinfo($name, PATHINFO_FILENAME);
    if($gridfile=="Grid"){
        $tmp_name = $_FILES['files']['tmp_name'][$f];
        move_uploaded_file($tmp_name, "$finalname/$name");
    } else {
        $tmp_name = $_FILES['files']['tmp_name'][$f];
        $newname = 'velbn_'.$i.'.dat';
        move_uploaded_file($tmp_name, "$finalname/$newname");
    //rename
        $i++;
    }
    //End of else
}
//End of foreach

//Shell Command
$cmd = "/Run.sh $finalname &2>1";
shell_exec($cmd);
//End of Shell

header("location:Index.php?img=citerweb/$finalname/$finalname.jpg");
?>
APPENDIX C

SHELL SCRIPT RUN.SH
#!/bin/sh

```
cd MatlabSourceCode
matlab -nodesktop -nojvm -nosplash -r "LCS1('$1') ; quit"
```
APPENDIX D

MATLAB SCRIPT LCS1.M
% close all, clear all, clc, pause on
addpath ProblemFiles
addpath ..
addpath ..\binary_files

global Xg Zg

% *************************** FTLE parameters ***************************

startStep = 16000;
LCS.t = 0.25; % time interval
LCS.particleDt = 1;
LCS.advectionType = 2;
LCS.problem = 'Cornell';
LCS.currentStep = 1;
LCS.currentTime = 0;
LCS.dt = 1.121566547803710e-004;
LCS.direction = 1; % +1 = forward, -1 = backward
LCS.skip = 4;

% *************************** Construct domain ***************************

fprintf('Constructing domain...
')

% Read data from velocity file
fname = 'velbn_1.dat';
fp = fopen(fname);
head = fread(fp,6,'*int32');
noOfNodes = head(2)*head(3)*head(4);

% Read mesh data from Excel file
head2 = xlsread('X-Z_Grid.xls','A1:D2');
meshdat = xlsread('X-Z_Grid.xls','A3:B num2str(noOfNodes+2));

% Construct mesh
Xg = reshape(meshdat(:,1),head(2),head(4));
Zg = reshape(meshdat(:,2),head(2),head(4));

Xg = Xg(:,1:head(4)/2);
Zg = Zg(:,1:head(4)/2);

fclose(fp)

% Initialize FTLE particles

fprintf('Initializing FTLE particles...
')
xMin = min(min(Xg));
xMax = max(max(Xg));
zMin = min(min(Zg));
zMax = max(max(Zg));
Lx = xMax-xMin;
Lz = zMax-zMin;

nz = 180;
nx = floor(nz*Lx/Lz);

[Xp1,Xp2] = meshgrid(xMin:Lx/nx:xMax,zMin:Lz/nz:zMax);
figure(1)
noOfParticles = (nx+1)*(nz+1);

% Allocate velocity
Vp{1} = 0*Xp{1};
Vp{2} = 0*Xp{1};
Vp{3} = 0*Xp{1};
Vp{4} = 0*Xp{1};
Vp{5} = 0*Xp{1};
Vp{6} = 0*Xp{1};

% Allocate dummy flow map for periodic conditions
XpDum{1} = 0*Xp{1};
XpDum2{1} = 0*Xp{1};
XpDiff{1} = 0*Xp{1};
XpDum{2} = 0*Xp{2};
XpDum2{2} = 0*Xp{2};
XpDiff{2} = 0*Xp{2};

%%%%% ************************************* Advect particles *************************************
fprintf('Advecting FTLE particles...\n')

% Set up time stepping
finalTime = LCS.t;
nSteps = floor(finalTime/LCS.dt);
sCounter = startStep*LCS.direction*LCSskip:startStep+LCS.direction*nSteps;
LCS.dt = LCS.dt*LCSskip;

% Advect particles using 3rd-order time integration
tic
for n = sCounter
    n
    XpDum{1} = Xp{1};
    XpDum{2} = Xp{2};

    % Apply periodic boundary conditions
    XpDum{1}(Xp{1}>Lx) = mod(Xp{1}(Xp{1}>Lx),Lx);
    XpDum{1}(Xp{1}<0) = Lx - mod(abs(Xp{1}(Xp{1}<0)),Lx);
    XpDum{2}(Xp{2}>Lz) = Lz;
    XpDum{2}(Xp{2}<0) = 0;

    % Advect particles
tic
    LoadVelocity(n);
toc
tic
[XpDum2, Vp, LCS] = AdvectParticlesSimple(XpDum, Vp, LCS);
toc

% Update positions
XpDiff{1} = XpDum2{1} - XpDum{1};
Xp{1} = Xp{1} + XpDiff{1};
XpDiff{2} = XpDum2{2} - XpDum{2};
\begin{verbatim}
end
toc

%%% ***************************** Compute FTLE ****************************

% Determine initial grid
\texttt{[Xpi{1},Xpi{2}] = meshgrid(xMin:Lx/nx:xMax,zMin:Lz/nz:zMax);};

\texttt{gSize = size(Xp{1})}
\texttt{Xp{1}(isnan(Xp{1})) = Xpi{1}(isnan(Xp{1}));}
\texttt{Xp{2}(isnan(Xp{2})) = Xpi{2}(isnan(Xp{2}));}

% Plot flow maps
\texttt{figure}
\texttt{contourf(Xpi{1},Xpi{2},Xp{1},80,’LineStyle’,’None’)}
\texttt{daspect([1 1 1])}
\texttt{title(’X-map’)}
\texttt{figure}
\texttt{contourf(Xpi{1},Xpi{2},Xp{2},80,’LineStyle’,’None’)}
\texttt{daspect([1 1 1])}
\texttt{title(’Y-map’)}

% Determine deformation gradient
\texttt{dXdx = (Xp{1}(;,:gSize{2}(:,:gSize{2}-2)))/(Lx/nx);};
\texttt{dYdx = (Xp{2}(;,:gSize{2}(:,:gSize{2}-2)))/(Lx/nx);};
\texttt{dXdy = (Xp{1}(;gSize{1},:)-(Xp{1}(;1:gSize{1}-2,:)))/(Lz/nz);};
\texttt{dYdy = (Xp{2}(;gSize{1},:)-(Xp{2}(;1:gSize{1}-2,:)))/(Lz/nz);};

\texttt{FTLE = Xp{1}(2:gSize{1}-1,2:gSize{2}-1)*0;}
\texttt{for we = 1:gSize{1}-2}
  \texttt{for j = 1:gSize{2}-2}
    \texttt{defGrad = [dXdx(i+1,j) dXdx(i,j+1);...}
      \texttt{dYdx(i+1,j) dYdx(i,j+1)];}
    \texttt{defGrad(isnan(defGrad)) = 10;};
    \texttt{FTLE(i,j) = log(norm(defGrad))/LCS.t;}
  \texttt{end}
\texttt{end}

\texttt{figure}
\texttt{contourf(Xp{1}(2:gSize{1}-1,2:gSize{2}-1),Xp{2}(2:gSize{1}-1,2:gSize{2}-1),}
  \texttt{,’FTLE’,80,’LineStyle’,’None’)}
\texttt{daspect([1 1 1])}
\end{verbatim}
APPENDIX E

JAVASCRIPT CITERUI.JS
/**
 * jQuery Easing v1.3 - http://gsgd.co.uk/sandbox/jquery/easing/
 * Uses the built-in easing capabilities added in jQuery 1.1
 * to offer multiple easing options
 */

// t: current time, b: beginning value, c: change in value, d: duration
jQuery.easing["swing"] = jQuery.easing["swing"];

jQuery.extend( jQuery.easing, {
  def: 'easeOutQuad',
  swing: function (x, t, b, c, d) {
    //alert(jQuery.easing.default);
    return jQuery.easing[ jQuery.easing.def ](x, t, b, c, d);
  },
  easeInQuad: function (x, t, b, c, d) {
    return c*(t/=d)*t + b;
  },
  easeOutQuad: function (x, t, b, c, d) {
    return -c * ((t=t/d)-2)*((t=t/d)-2) + b;
  },
  easeInCubic: function (x, t, b, c, d) {
    return c*(t/=d)*t*t + b;
  },
  easeOutCubic: function (x, t, b, c, d) {
    return c*((t=t/d)-1)*t*t + b;
  },
  easeInOutCubic: function (x, t, b, c, d) {
    if ((t=t/d) < 1) return c/2*t*t*t + b;
    return c/2 * ((1-t)*(t-2) - 1) + b;
  },
  easeInQuart: function (x, t, b, c, d) {
    return c*(t/=d)*t*t + b;
  },
  easeOutQuart: function (x, t, b, c, d) {
    return -c * ((t=t/d)-2)*((t=t/d)-2) + b;
  },
  easeInOutQuart: function (x, t, b, c, d) {
    if ((t=t/d) < 1) return c/2*t*t*t*t + b;
    return c/2 * ((t=t/2)-2)*t*t*t + b;
  },
  easeInQuint: function (x, t, b, c, d) {
    return c*(t/=d)*t*t*t + b;
  },
  easeOutQuint: function (x, t, b, c, d) {
    return c*((t=t/d)-1)*t*t*t + b;
  },
  easeInOutQuint: function (x, t, b, c, d) {
    if ((t=t/d) < 1) return c/2*t*t*t*t*t + b;
    return c/2 * ((t=t/2)-2)*t*t*t*t + b;
});
easeInSine: function (x, t, b, c, d) {
    return -c * Math.cos(t/d * (Math.PI/2)) + c + b;
}

easeOutSine: function (x, t, b, c, d) {
    return c * Math.sin(t/d * (Math.PI/2)) + b;
}

easeInOutSine: function (x, t, b, c, d) {
    return -c/2 * (Math.cos(Math.PI*t/d) - 1) + b;
}

easeInExpo: function (x, t, b, c, d) {
    return (t == 0) ? b : c * Math.pow(2, 10 * (t / d) - 1) + b;
}

easeOutExpo: function (x, t, b, c, d) {
    return (t == d) ? b+c : c * (-Math.pow(2, -10 * t / d) + 1) + b;
}

easeInOutExpo: function (x, t, b, c, d) {
    if (!t || !d) return b;
    if (t == d) return b+c;
    if (t/d < 1) return c/2 * Math.pow(2, 10 * (t - 1)) + b;
    return c/2 * (-Math.pow(2, -10 * t - 1) + 2) + b;
}

easeInCirc: function (x, t, b, c, d) {
    return -c * (Math.sqrt(1 - (t/d)^2) - 1) + b;
}

easeOutCirc: function (x, t, b, c, d) {
    return c * Math.sqrt(1 - (t/d-1)^2) + b;
}

easeInOutCirc: function (x, t, b, c, d) {
    if ((t/d) < 1) return -c/2 * (Math.sqrt(1 - t^2) - 1) + b;
    return c/2 * (Math.sqrt(1 - (t-2)^2) + 1) + b;
}

easeInElastic: function (x, t, b, c, d) {
    var s = 1.70158; var p = 0; var a = c;
    if (t == 0) return b; if ((t < d) return b+c; if (t == d) p = d/3;
    if (a < Math.abs(c)) { a = c; var s = p/4; }
    var s = p/(2*Math.PI) * Math.asin(s/c); return -a*Math.pow(2,10*(t==1)) * Math.sin((t*d-s)*(2*Math.PI/p) ) + b;
}

easeOutElastic: function (x, t, b, c, d) {
    var s = 1.70158; var p = 0; var a = c;
    if (t == 0) return b; if ((t < d) return b+c; if (t == d) p = d*3;
    if (a < Math.abs(c)) { a = c; var s = p/4; }
    var s = p/(2*Math.PI) * Math.asin(s/c); return a*Math.pow(2,-10*t) * Math.sin((t*d-s)*(2*Math.PI/p) ) + c + b;
}

easeInOutElastic: function (x, t, b, c, d) {
    var s = 1.70158; var p = 0; var a = c;
    if (t == 0) return b; if ((t < d/2) == 2) return b+c; if (t == d) p = d*(3*1.5);
    if (a < Math.abs(c)) { a = c; var s = p/4; }
    var s = p/(2*Math.PI) * Math.asin(s/c);
    if (t < 1) return -0.5*(a*Math.pow(2,10*(t==1)) * Math.sin( (t*d-s)*(2*Math.PI/p) ) + b;
    return a*Math.pow(2,-10*(t==1)) * Math.sin( (t*d-s)*(2*Math.PI/p) ) * 0.5 + c + b;
}

easeInBack: function (x, t, b, c, d, s) {
    if (s == undefined) s = 1.70158;
}
return c*(t/d)*t*((s+1)*t - s) + b;
},
easeOutBack: function (x, t, b, c, d, s) {
  if (s == undefined) s = 1.70158;
  return c*((t/t/d-1)*t*((s+1)*t + s) + 1) + b;
},
easeInOutBack: function (x, t, b, c, d, s) {
  if (s == undefined) s = 1.70158;
  if ((t/d/2 < 1) return c/2*(t*t*((s*=(1.525))+(1)*t - s)) + b;
  return c/2*((t==2)*t*(s==(1.525))+(1)*t + s) + 2) + b;
},
easeInBounce: function (x, t, b, c, d) {
  return c - jQuery.easing.easeOutBounce (x, d-t, 0, c, d) + b;
},
easeOutBounce: function (x, t, b, c, d) {
  if (t < d/2) return jQuery.easing.easeOutBounce (x, t, d, 0, c, d) + .5 + b;
  return jQuery.easing.easeOutBounce (x, t, d, 0, c, d) + .5 + c*.5 + b;
});
APPENDIX F

CSS CITERCSS.CSS