**Case Study:**

**Website for Fernandez Translation Services**

**Dayana Fernandez**

**Towson University**

**AIT 715**

**Dr. Josh Dehlinger**

**May 5, 2018**

Table of Contents

I. Abstract 2

II. Introduction 3

III. Technology & Terminology 4

IV. Architecture 8

V. Use Cases 9

VI. System Components 10

VII. Website 17

VIII. Challenges & Lessons Learned 17

IX. Future plans 19

X. Source Code 19

XI. Acknowledgements 20

XII. Reading List/Bibliography/Works Cited 20

## I. Abstract

The purpose of this case study project was to design and develop a website to promote translation services. I wanted a website to provide information and advertise my services in written translations from English to Spanish and Spanish to English. Customers will be able to get in touch with me to request services and get an estimated cost for their translation. Furthermore, they will be notified of the approximate time of completion. The major objective for this project was to develop a website that will work on all web browsers. Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), JavaScript (JS), Hypertext Preprocessor (PHP) and Structured Query Language (SQL) were used for the development of this website. I used GitHub as the software repository. This has an impact because although I have done a few translations, I don’t currently have a website that allows me to advertise my services. One of my goals is to get professionally certified so that I can expand my abilities and be able to reach out to a broader audience, so this website will also aid me in making this happen.

## II. Introduction

The project is a website that will advertise my translation services. I will offer written translation services both from English into Spanish and Spanish into English. The scope of this project are be the requirements, design and development of a website that will work on multiple web browsers.

The objective and main outcome of this project is the delivery of a fully functional and publicly available website that allows customers to elicit my services and allow me to interact with them as well. The first deliverable was a proposal in order to be granted permission to work on the project. The second deliverable was a prototype of the initial look of the site in order to make sure that it meets all required web standards. The subsequent deliverables were the individual pages that make up the overall site. The final deliverable was the live and functional site. This case study is suitable because it aligns with all the concepts taught in Fundamentals of Web Technologies and Development. I used the knowledge gained in the course in order to design the site and make sure that I followed the required web standards. I also had to tap into the knowledge gained in Client/Server-Side Programming on the Web in order to keep back-end requirements in mind as I was designing.

All of the above learning elements from this project are suitable because they allowed me to make a website that will be used to provide services and use the finished product as something tangible to show my capabilities during potential job interviews. I think it is be suitable as a capstone project because the final product has seven pages, which are be dynamic and interactive, which adds to the complexity. All of the elements and nuances were carefully thought-out and planned in order to provide the most complete site possible.

## III. Technology & Terminology

The following is a list of technology that I used for each aspect of this project.

**Hypertext Markup Language** (**HTML**) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997 [9].

**Cascading Style Sheets** (**CSS**) is a style sheet language used for describing the presentation of a document written in a markup language like HTML.CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .CSS file, and reduce complexity and repetition in the structural content. Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device. The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C).. The W3C operates a free CSS validation service for CSS documents [3].

**JavaScript (JS)** is a high-level, interpreted programming language. It is a language, which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm. Alongside HTML and CSS, JavaScript is one of the three core technologies of the World Wide Web. JavaScript enables interactive web pages and thus is an essential part of web applications. The vast majority of websites use it, and all major web browsers have a dedicated JavaScript engine to execute it. As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles. It has an API for working with text, arrays, dates, regular expressions, and basic manipulation of the DOM, but the language itself does not include any I/O, such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded. Initially only implemented client-side in web browsers, JavaScript engines are now embedded in many other types of host software, including server-side in web servers and databases, and in non-web programs such as word processors and PDF software, and in runtime environments that make JavaScript available for writing mobile and desktop applications, including desktop widgets [10].

**Hypertext Preprocessor** (**PHP**) is a server-side scripting language designed for web development but also used as a general-purpose programming language. It was originally created by Rasmus Lerdorf in 1994. The PHP reference implementation is now produced by The PHP Group. PHP originally stood for Personal Home Page, but it now stands for the recursive acronym PHP: Hypertext Preprocessor. PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems, and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications. The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge. The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014 work has gone on to create a formal PHP specification. During the 2010s there have been increased efforts towards standardization and code sharing in PHP applications by projects such as PHP-FIG in the form of PSR-initiatives as well as Composer dependency manager and the Packagist repository. PHP hosts a diverse array of web frameworks requiring framework-specific knowledge, with Laravel recently emerging as a popular option by incorporating ideas made popular from other competing non-PHP web frameworks, like Ruby on Rails [15].

**Structured Query Language (SQL)** is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). It is particularly useful in handling structured data where there are relations between different entities/variables of the data. In comparison to older read/write APIs like ISAM or VSAM, SQL offers two main advantages: first, it introduced the concept of accessing many records with one single command; and second, it eliminates the need to specify how to reach a record, e.g. with or without an index. Originally based upon relational algebra and tuple relational calculus, SQL consists of many types of statements, which may be informally classed as sublanguages, commonly: a data query language (DQL), a data definition language (DDL), a data control language (DCL), and a data manipulation language (DML). The scope of SQL includes data query, data manipulation (insert, update and delete), data definition (schema creation and modification), and data access control. Although SQL is often described as, and to a great extent is, a declarative language (4GL), it also includes procedural elements. SQL was one of the first commercial languages for Edgar F. Codd's relational model, as described in his influential 1970 paper, "A Relational Model of Data for Large Shared Data Banks". Despite not entirely adhering to the relational model as described by Codd, it became the most widely used database language. SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987. Since then, the standard has been revised to include a larger set of features. Despite the existence of such standards, most SQL code is not completely portable among different database systems without adjustments [21].

**GitHub** (originally known as Logical Awesome LLC) is a web-based hosting service for version control using git. It is mostly used for computer code. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project. GitHub offers plans for both private repositories and free accounts, which are commonly used to host open-source software projects. As of April 2017, GitHub reports having almost 20 million users and 57 million repositories, making it the largest host of source code in the world [8].

The term **user**, as referenced in this document, refers to anyone who visits the website, regardless of whether or not, they create an account. Users have the option of simply inquiring about services, without being obligated to create an official account. On the other hand, once someone creates an account, they become a **customer**, even if they don’t request services right away. **Administrator** will also be referenced, because even though this only refers me at the moment, in the future, if more people join me, then they will also be given administration privileges.

## IV. Architecture

The framework for the website is HTML 5, CSS, PHP, JavaScript and SQL. These will be used in order to make the website dynamic and interactive, in order to provide visitors with the best and most effortless experience possible. HTML 5 will serve as the main framework, while JavaScript and CSS will provide the enhancements and dynamism. PHP and SQL will serve on the client/server backend. This will make it easy and seamless for users to navigate, regardless of their level of experience using the World Wide Web.

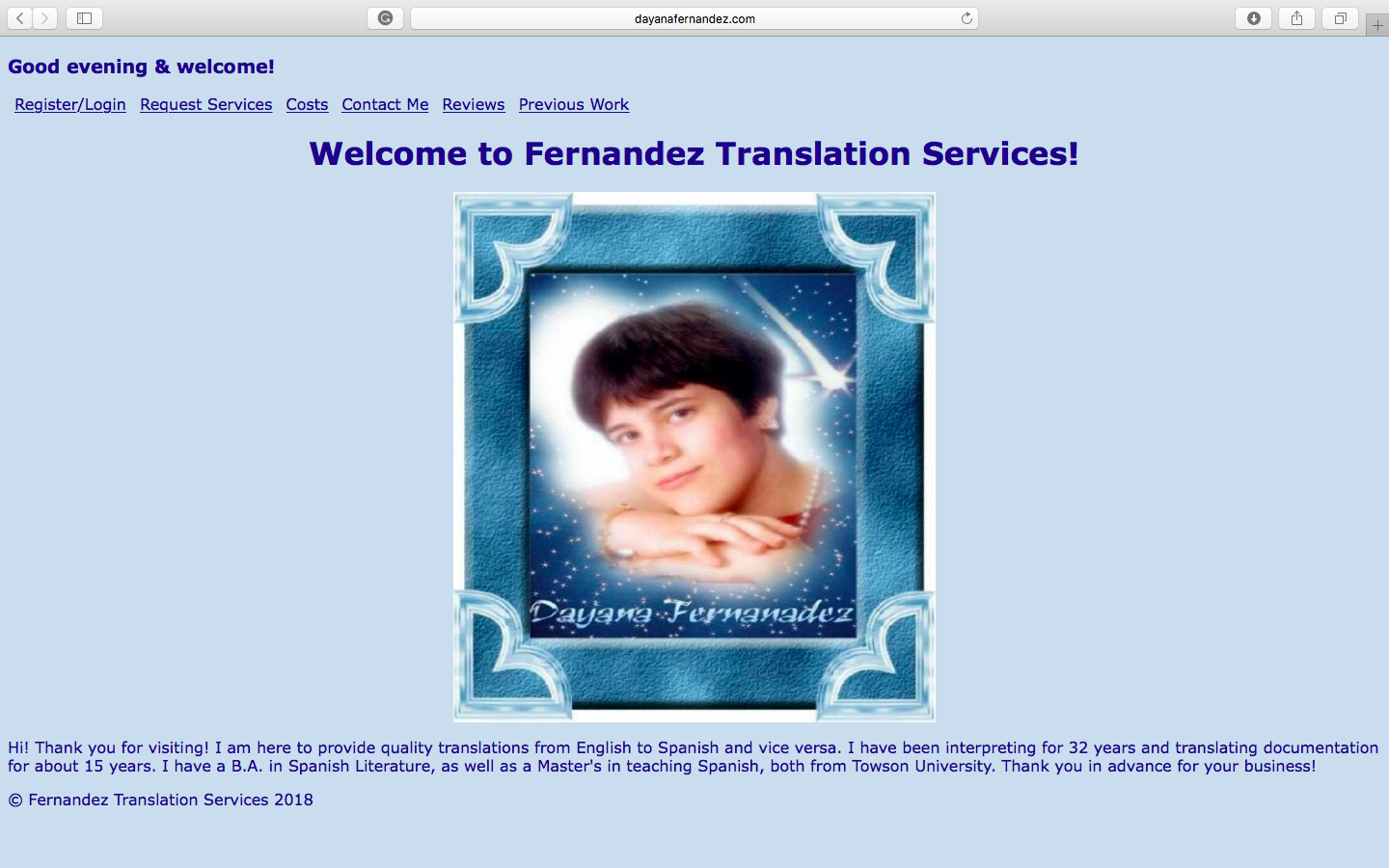


## V. Use Cases

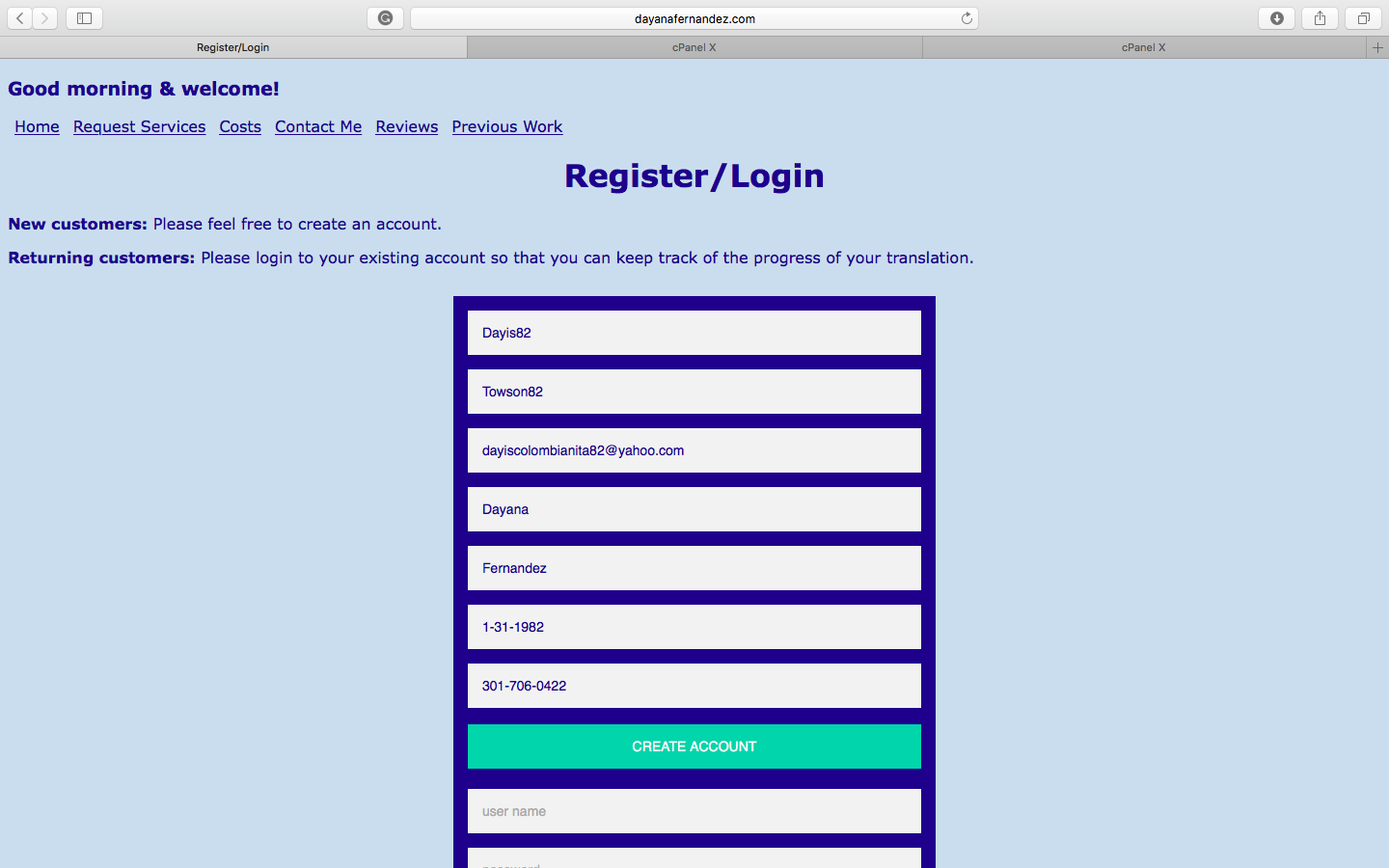
1. The system shall allow users to create an account.
2. The system shall allow users to request services.
3. The system shall allow users to see an estimated cost of service.
4. The system shall allow users to see a timeline of service completion.
5. The system shall allow users to contact me with any questions.
6. The system shall allow users to post reviews on services received.
7. The system shall allow storage of a repository of original texts.
8. The system shall allow storage of a repository of translated material.
9. The system shall allow documentation to be sent and received.

## VI. System Components

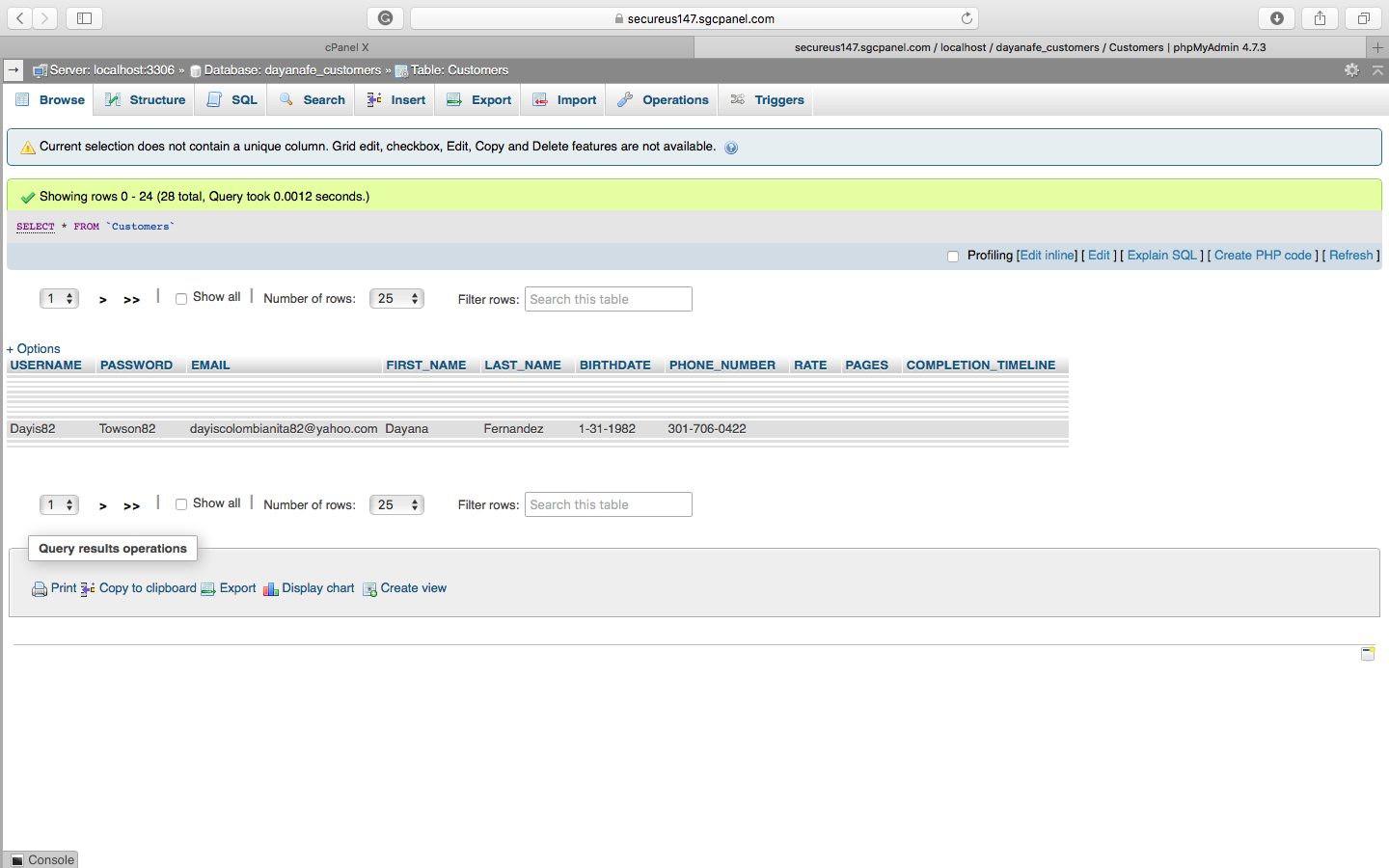
This is the home page that has a blurb about me and what the site is about. Both customers and users can access this page.



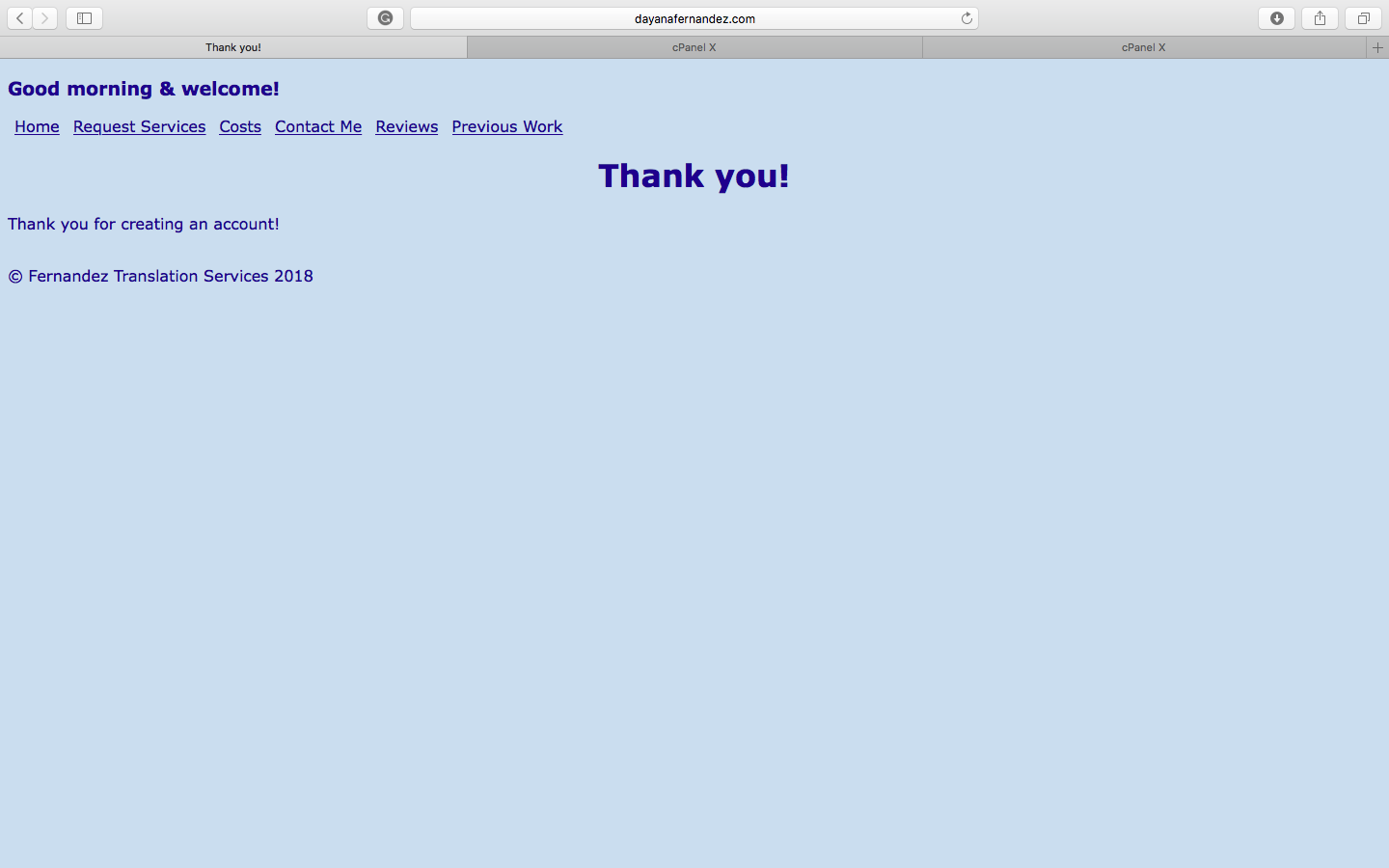
This is the Register screen where customers can create an account in order to get translations done.



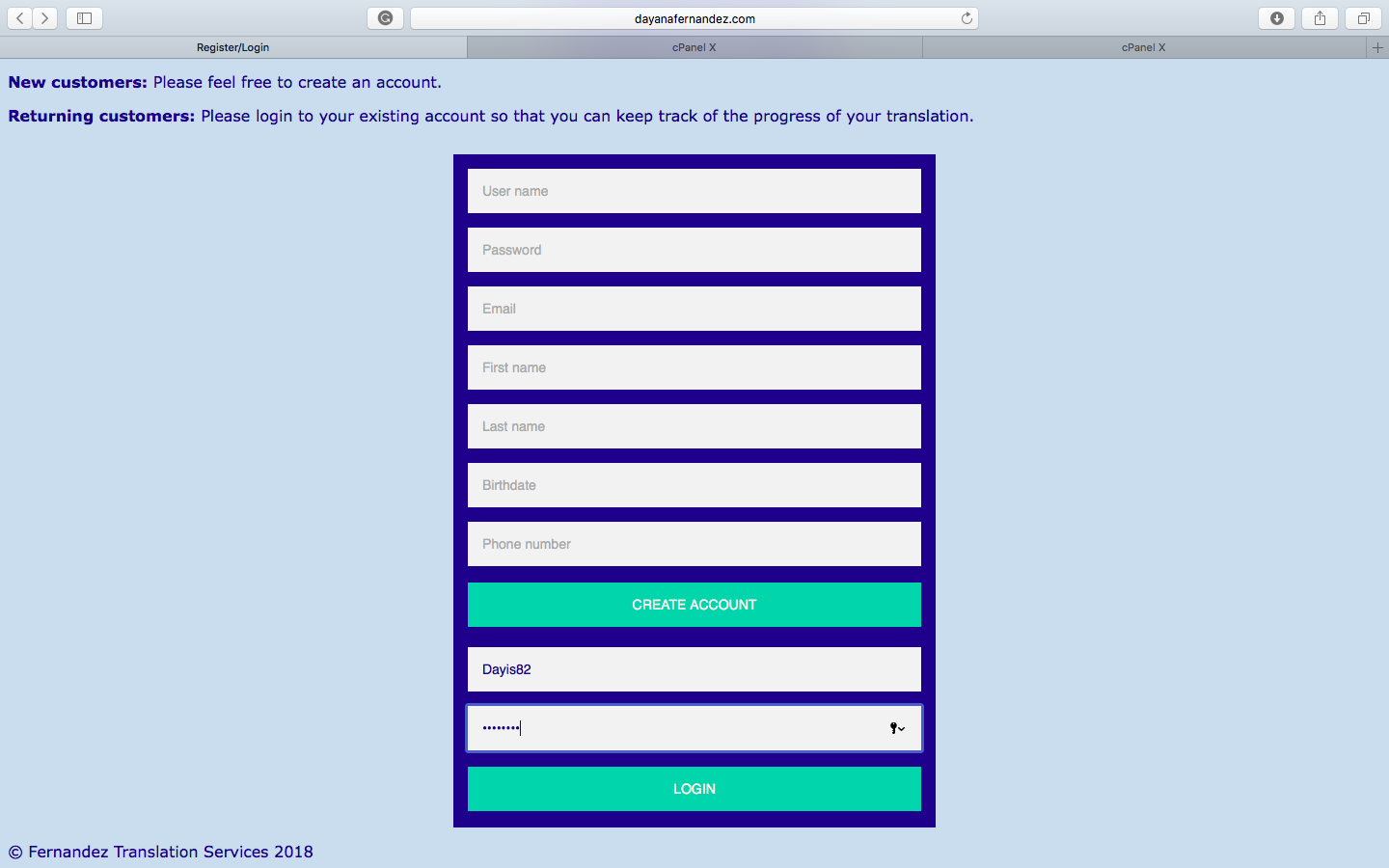
This is the database view that I am showing here, but obviously this is not something that users or customers will see. The last three columns will be filled in once that information is available.



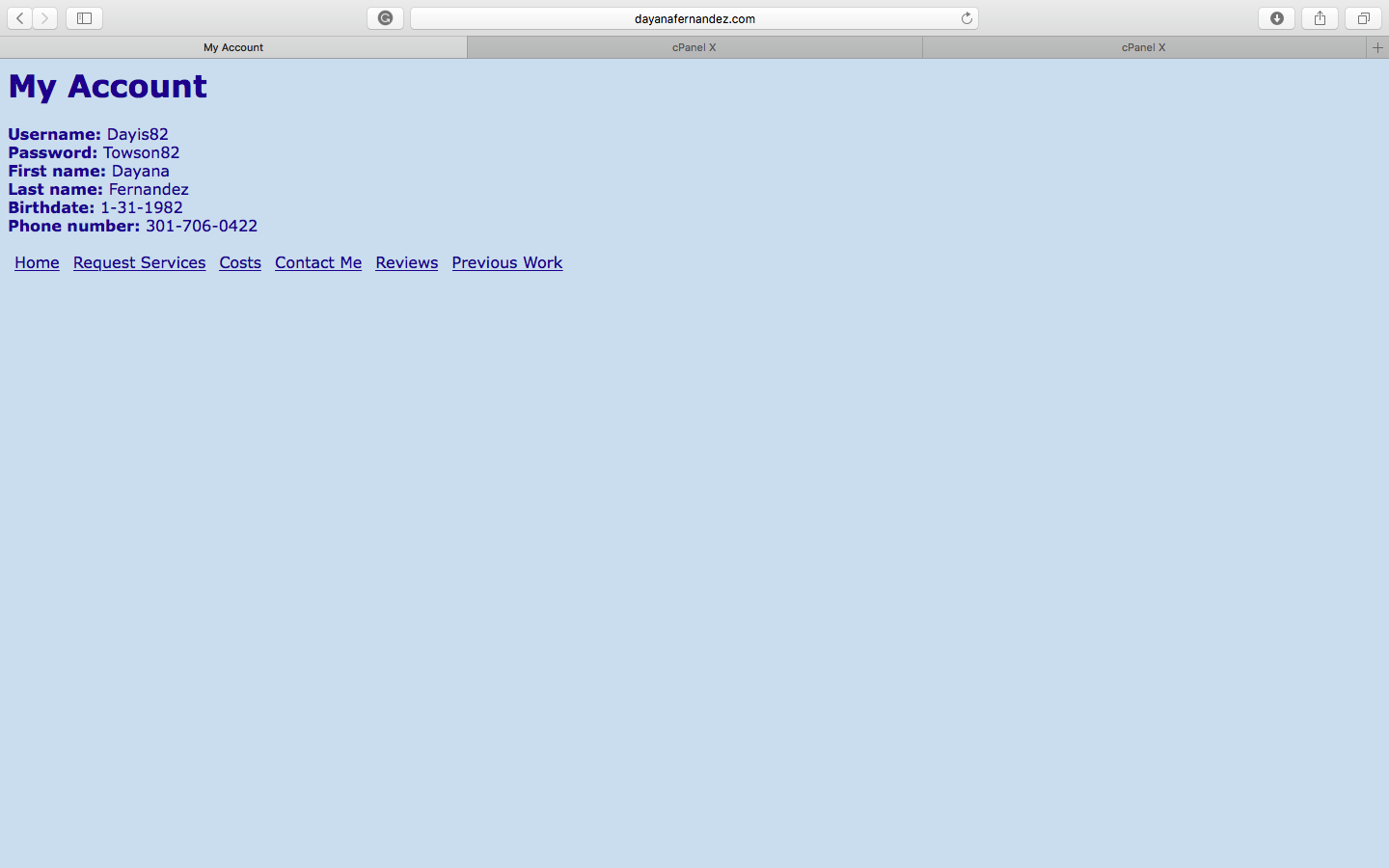
This is the page that customers will see once they click the “Create Account” button so that they know that they are now an official customer. (If an account is not created, whoever visits the site will be considered users and they will still be able to navigate certain parts of the site.



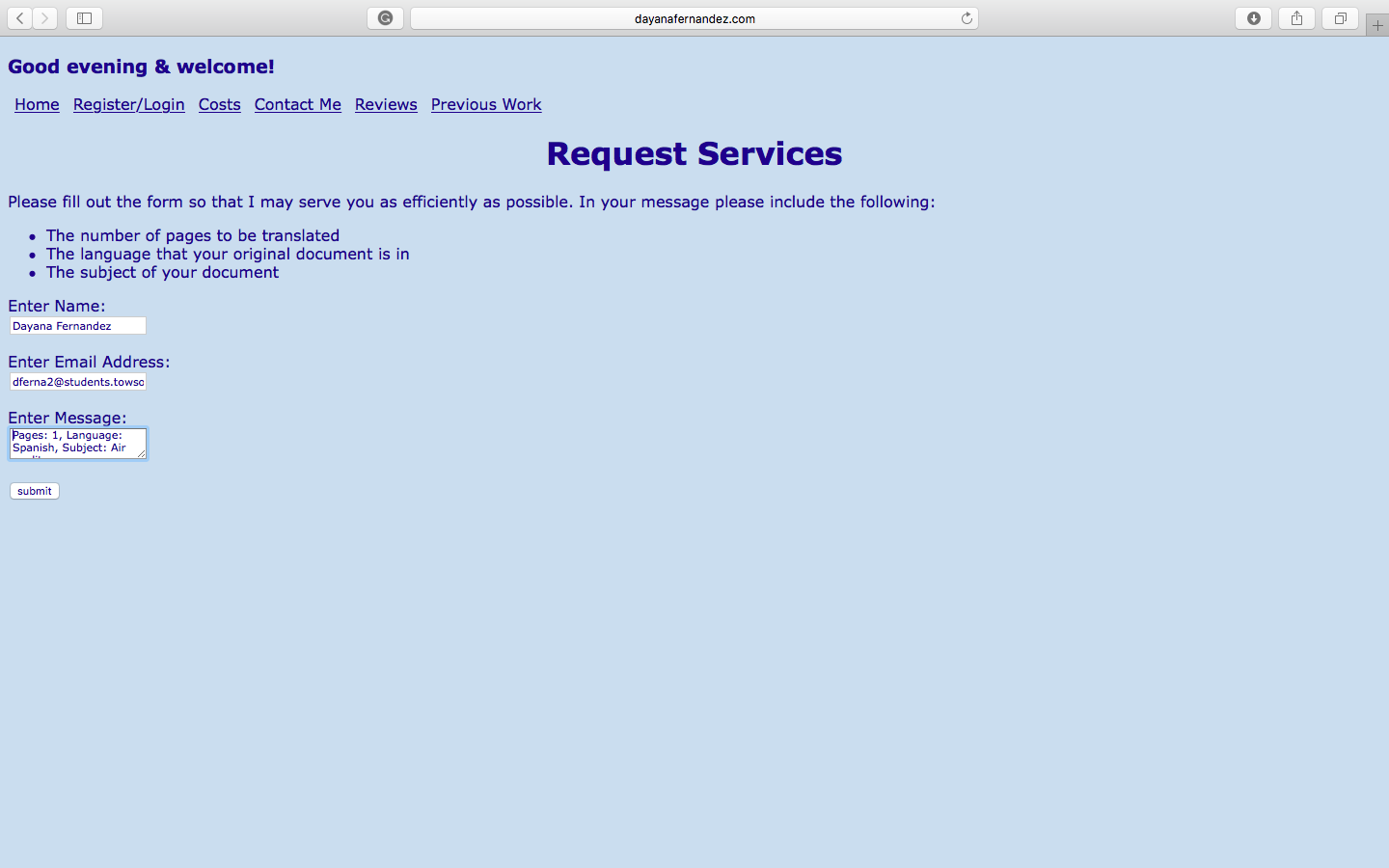
This would be the “Login” portion



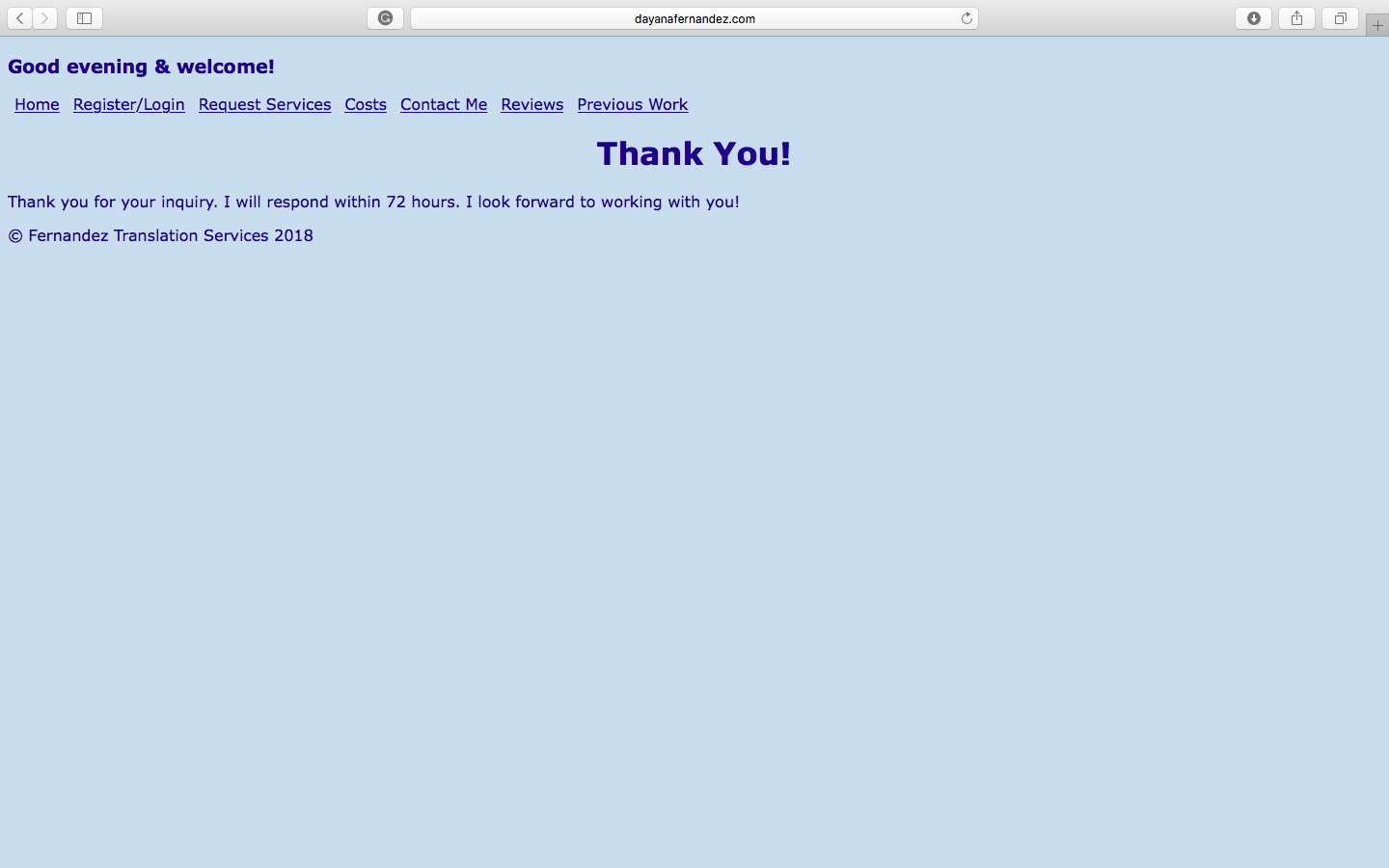
This would be the page that would appear once the customer presses the login button. Obviously once the other information is added to the database, the appropriate information will be added here.



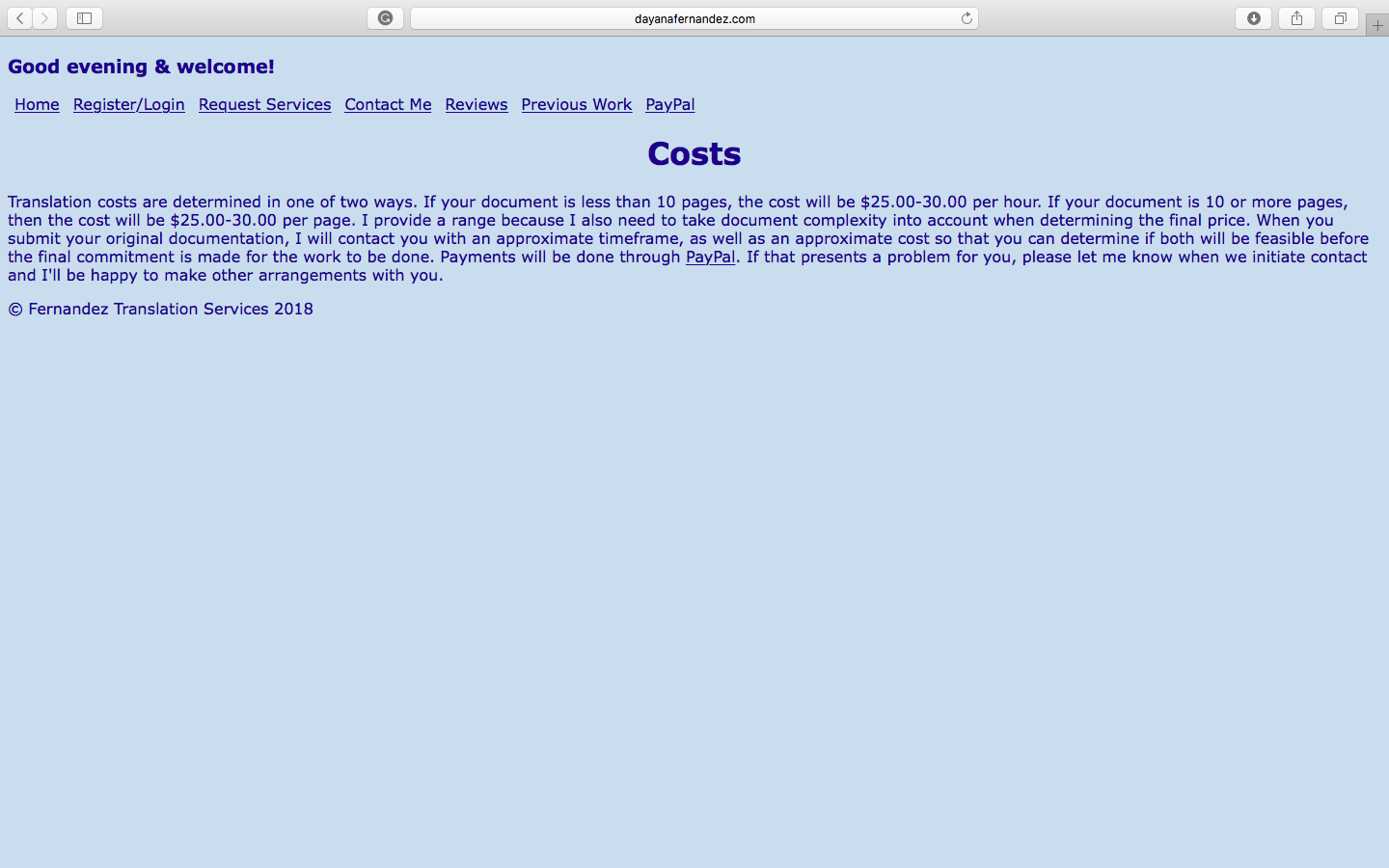
This is what is seen when the “Request Services” link is clicked. When the form is filled out, clicking submit will send the information to me in an email. This page can be accessed by both users and customers because they can request or inquire about services without creating an account.



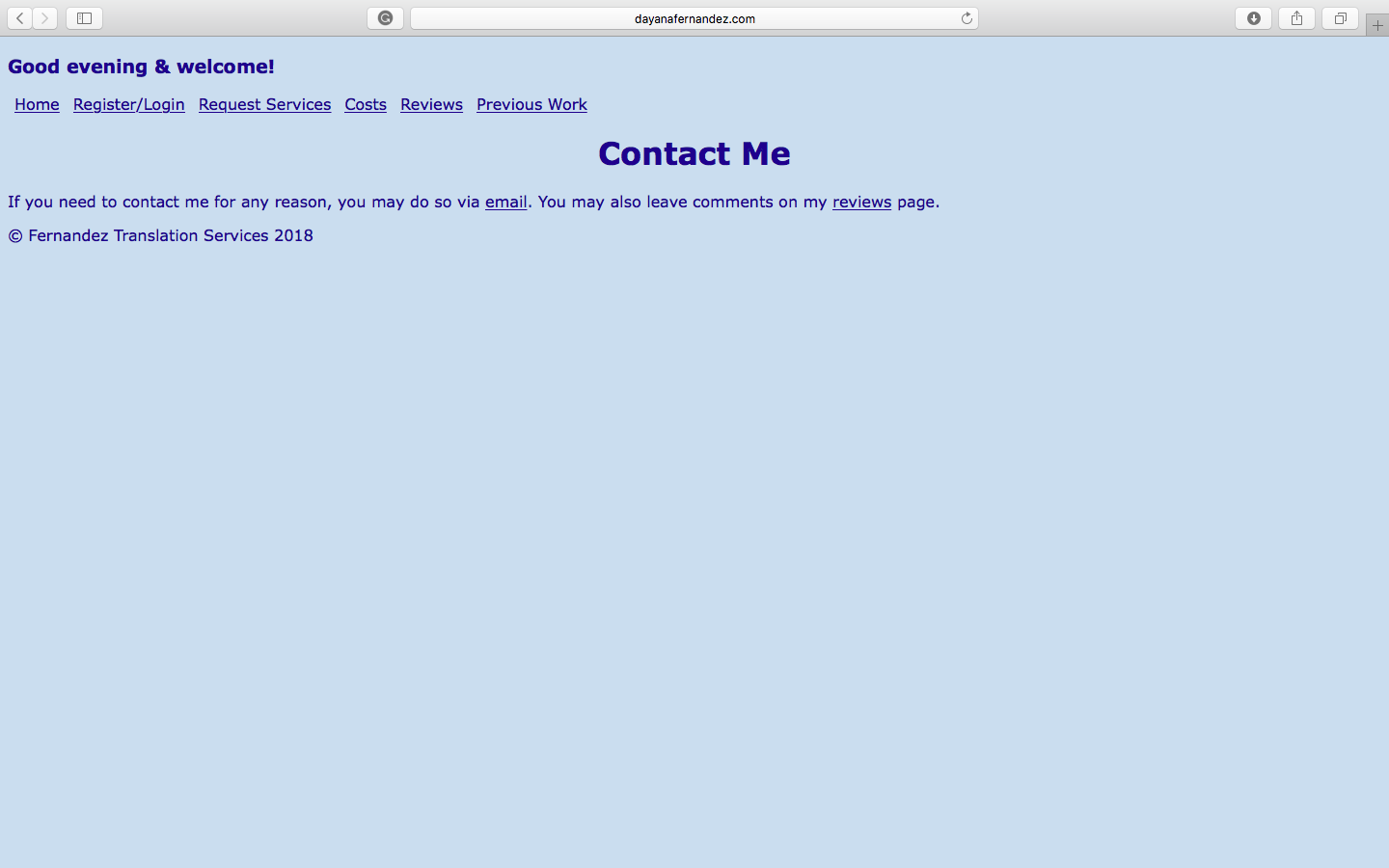
This is the page that customers see once the submit button is clicked.



The is the page that is seen when the costs link is clicked. This also has a paypal link. Users and customers can see this page.



The is the page that goes to the contact me link. When the email link is clicked, an email client opens in order to be able to email directly inside it. Both customers and users can see this page.



This page appears when the reviews link is clicked and it allows users to post reviews on services received. Users can read reviews and customers can read and post reviews.



This page appears when the previous work link is clicked. The links within this page go to PDF documents that customers and users can read. This is a page that can be accessed with or without creating an account.



## VII. Website

The live site can be accessed at the following address: <http://www.dayanafernandez.com>

## VIII. Challenges & Lessons Learned

The biggest and #1 lesson that I learned was to be prepared to learn new technologies as you are working! Things don’t always turn out the way you expect them to and the plans that you have laid out, no matter how detailed and careful you are, will most likely need to be changed or even scratched all together! Allow yourself time for those unexpected issues so that you don’t end up short-changing yourself!

When I started this project, I thought I was going to use HTML, CSS, JavaScript, Node JS and MongoDB, as those were the technologies that I had been taught in the AIT classes. I had also figured that I would use Amazon Web Services to deploy the live site. But I realized that TU gives students space on the tiger server so I decided to deploy the site using the tiger server.

The front-end development aspects went smoothly, but the client/server side was a different story. When I started developing the back-end, I tried to get Node JS and Mongo to work together with the site but nothing seemed to work correctly with the aspects that I had been taught and none of the database options I was coming across were working either. Finally, Dr. Dehlinger found a database option that seemed easy to work with, but then that wouldn’t work with the coding either. I then turned to YouTube and books to see if I could figure it out, but surprisingly there really isn’t that much-detailed information that combines how to pair Node and Mongo in order for them to work in conjunction with an HTML page.

I figured I would try and see if I could use AWS to take care of the database aspect and if I was able to set that up, then I would just transfer the HTML pages to Amazon’s server and deploy everything from there, but that didn’t work either because even though I tried the process three times, I’d get to a certain spot in database setup and I couldn’t get any further, no matter what I tried to do. At this point I was running out of options that I was familiar with so I was getting pretty desperate. Reluctantly I turned to the books I had bought on PHP & SQL web development. I was reluctant because I had bought them when I thought AIT 618 was going to be PHP based, but when things changed, I put them off to the side with the intention of starting to read them when I had “free time”. So really, the only exposure I had to PHP was 90 minutes of lecture from the end of AIT 616 and honestly, it was so confusing to me that the only thing that I knew for sure was that it was a server side language whose code could be embedded within HTML pages.

Now I was faced with the reality that if I wanted this site to really be useful, I had to learn this new language that not only intimidated the heck out of me, but I was on a time crunch to learn it, and the only thing I had going for me was the fact that I already knew SQL really well, which turned out to be the database aspect that I would need to use along with PHP. So I thought ok, I’ll be fine there because I would just ask Prof. Shasho to let me borrow a user name and password so I can use the classroom database and since the site is also on a TU server, that will work out fine. So, I’m working on PHP coding and I go to upload the scripts to the Tiger server, but again nothing seems to work, even though I’m following coding instructions carefully. I realized then that the Tiger server doesn’t support PHP!

AHHHHHH! Now what? So it was back to the drawing board with having to find a place to host the site. Here I ran into issues as well. Obviously I found plenty of sites, but those that were free either didn’t support PHP, or I needed to use their website builders and templates. Of course I was not about to let that happen after all the work I put in coding from scratch and the fact that I would have felt like I was cheating! I also found a few “free” ones that had PHP as a paid add-on. Finally, I found one (or rather I should say Dr. D found one) that seemed to have everything I needed for free, but when I went to click it, it gave me a message that said that the free server was full, so I needed to choose a paid server, so I finally gave in and made the decision to suck it up and pay, but when I tried to contact their customer service department to ask questions before purchasing, I got no response for about an hour, so I gave up on them.

FINALLY I came across siteground.com which is what I finally settled on because yes it is a paid option but their customer service and tech team is AMAZING and the set up process was so smooth and painless, it literally scared me and I kept waiting for the representative to tell me about some hidden catch, hahaha. But he patiently guided me through the entire process and I discovered I could have my own domain for as long as I kept the site with them. This whole ordeal just make me even more proud of the work I’ve done and the fact that I didn’t give up when things got tough, but rather just kept at it because I knew there had to be a solution. This is also a very important lesson, not only when working on a project, but also as a life lesson. Things aren’t always easy, but one should never give up because nothing that is worth it is easy and when you finally reach a goal that you’ve worked hard for, the victory is that much sweeter and you value it even more.

## IX. Future plans

In terms of future plans, I would like to make the site mobile-enabled. I also want to provide users the option of viewing the site in Spanish in order to accommodate users who don’t speak English. I would also like to add the services of interpretation, whether it is face-to-face or over the phone service, which would entail working with freelance interpretation companies.

## X. Source Code

All code is available at <https://github.com/DayanaFernandez/AIT-715>.

## XI. Acknowledgements

First, I’d like to thank Dr. Josh Dehlinger for his incredible patience and support throughout this entire process. Thank you so much Dr. D, as always for answering my endless questions and helping in any way you could. I truly could not have done this without you and would have lost it more than once, had it not been for you! But then again, this has been a constant since the moment I met you, so I should say, thank you for continuing to save my sanity and for your constant support, especially during the tough times during these almost 2 years! It has truly been a honor and a pleasure to be your student and advisee!

Next, thank you to Prof. Harry Shasho for allowing me to “borrow” a username and password for the classroom oracle database since I thought I was going to need to use it, since the website was originally launched on the TU tiger server and I thought it would be easier to get a database to connect if everything was on TU servers.

Finally, thank you, Andrea Chenowith, for providing me with lab times so I would have the option of working in the labs when I thought I would have to use the classroom database.

## XII. Reading List/Bibliography/Works Cited

[1] Beighley, L. & Morrison, M. (2009). Head First PHP & MySQL. O’Reiley Media, Inc. Sebastopol, California.

[2] Benedetti, R. & Cranley, R. (2011). Head First jQuery. O’Reiley Media, Inc. Sebastopol, California.

[3] “Cascading Style Sheets”, Wikipedia, Wikimedia Foundation, Inc. April 23, 2018. [Online.] Available: <https://en.wikipedia.org/wiki/Cascading_Style_Sheets>. [Accessed: April 29, 2018].

[4] Connolly, T. & Begg, C. (2015). Database Systems: A Practical Approach to Design, Implementation, and Management. Pearson Education, Inc. Upper Saddle River, New Jersey. Sixth Edition.

[5] Duckett, J. (2011). HTML & CSS: Design and Build Websites. John Wiley & Sons, Inc. Indianapolis, Indiana.

[6] Duckett, J. (2014). Javascript & JQuery: Interactive Front-end Web Development. John Wiley & Sons, Inc. Indianapolis, Indiana.

[7] Freeman, E. & Robson, E. (2011). Head First HTML5 Programming: Building Web Apps with JavaScript. O’Reiley Media, Inc. Sebastopol, California.

[8] “GitHub”, Wikipedia, Wikimedia Foundation, Inc. April 25, 2018. [Online.] Available: <https://en.wikipedia.org/wiki/GitHub>. [Accessed: April 29, 2018].

[9] “HTML”, Wikipedia, Wikimedia Foundation, Inc. April 24, 2018. [Online.] Available: <https://en.wikipedia.org/wiki/HTML>. [Accessed: April 29, 2018].

[10] “JavaScript”, Wikipedia, Wikimedia Foundation, Inc. April 27, 2018. [Online.] Available: <https://en.wikipedia.org/wiki/JavaScript>. [Accessed: April 29, 2018].

[11] Karlins, D. & Sahlin, D. (2012). Building Websites All-in-one for Dummies. John Wiley & Sons, Inc. Hoboken, New Jersey. Third Edition.

[12] McFedries, P. (2008). The Complete Idiot’s Guide to Creating a Website. Alpha: Penguin Group, Inc. New York, New York.

[13] Mihajlov, M. (2015). HTML Quickstart Guide: The Simplified Beginner’s Guide to HTML. ClydeBank Media LLC. Albany, New York.

[14] Minnick, C. & Holland, E. (2015). Coding with Javascript for Dummies. John Wiley & Sons, Inc. Hoboken, New Jersey.

[15] “PHP”, Wikipedia, Wikimedia Foundation, Inc. April 27, 2018. [Online.] Available: <https://en.wikipedia.org/wiki/PHP>. [Accessed: April 29, 2018].

[16] Robson, E. & Freeman, E. (2012). Head First HTML and CSS: A Learner's Guide to Creating Standards-Based Web Pages. O’Reiley Media, Inc. Sebastopol, California. Second Edition.

[17] Robson, E. & Freeman, E. (2014). Head First Javascript Programming. O’Reiley Media, Inc. Sebastopol, California.

[18] Sabin-Wilson, L. (2017). WordPress All-in-One for Dummies. John Wiley & Sons, Inc. Hoboken, New Jersey. Third Edition.

[19] Siarto, J. (2010). Head First WordPress. O’Reiley Media, Inc. Sebastopol, California.

[20] Sommerville, I. (2011). Software Engineering. Addison-Wesley. Boston, Massachusetts. Ninth Edition.

[21] “SQL”, Wikipedia, Wikimedia Foundation, Inc. April 26, 2018. [Online.] Available: <https://en.wikipedia.org/wiki/SQL>. [Accessed: April 29, 2018].

[22] Valade, J., Ballad, T. & Ballad, B. (2008). PHP & MySQL Web Development: All-in-One Desk Reference for Dummies. John Wiley & Sons, Inc. Hoboken, New Jersey.

[23] Watrall, E. & Siarto, J. (2009). Head First Web Design: A Learner's Companion to Accessible, Usable, Engaging Websites. O’Reiley Media, Inc. Sebastopol, California.

[24] Welling, L. & Thompson, L. (2009). PHP and MySQL Web Development. Pearson Education, Inc. Upper Saddle River, New Jersey. Fourth Edition.