Northwest Nazarene University

Styling Edits for the Northwest Nazarene University Application Status Portal

THESIS

Submitted to the Department of Mathematics and Computer Science

in partial fulfillment of the requirements

for the degree of

BACHELOR OF ARTS

Nathan Sage Musaku Mwiinga

2021

THESIS

Submitted to the Department of Mathematics and Computer Science

in partial fulfillment of the requirements

for the degree of

BACHELOR OF ARTS

By

Nathan Sage Musaku Mwiinga

2021

Styling Edits for the Northwest Nazarene University Application Status Portal

Author: ______________________________
Nathan Sage Musaku Mwiinga

Approved: Dale A Ham ______________________________

Dale Hamilton, Computer Science Ph.D., Associate Professor, Department of Mathematics and Computer Science, Faculty Advisor

Approved: Sage Mwiinga ______________________________
Sage Mwiinga, CRM Manager
Second Reader

Approved: Barry Myers ______________________________
Barry L. Myers, Ph.D., Chair,
Department of Mathematics and Computer Science

Abstract

Styling Edits for the Northwest Nazarene University Application Status Portal.

MWIINGA, NATHAN (Department of Mathematics and Computer Science), HAMILTON, DR. DALE (Department of Mathematics and Computer Science), MYERS, DR. BARRY (Department of Mathematics and Computer Science).

The Application Status Portal is a tool within Slate, the Customer Relationship Management (CRM) system used by the admissions teams of Northwest Nazarene University (NNU) and several other universities, to assist students in the application process by providing information about needed, completed, and or missing documentation and forms. Slate was introduced three years ago at NNU and has seen work from the university's CRM manager, Sage Mwiinga, in the setup and configuration for the use of many individuals from the university. The application status portal has not seen many updates in the realm of styling since it was first implemented in the summer of 2018. Because of that, elements of the portal do not contain a consistent styling or culture across the portal. The purpose of this project was to utilize CSS classes to add design elements to make the portal stand out while also being similar in concept to the main NNU website for users on various device types. The resulting portal cultivated a familiar feel to the nnu.edu domain using positioning and brand colors that shows relation and connection throughout the portal and to the NNU website. Future work on the portal will see the need of implementing Tailwind CSS, a CSS framework that is better suited for making responsive websites.

Acknowledgements

I would like to express a profound thank you to my parents, Sage and Lorraine, for the sacrifices you have made over the years for Natasha and me, I know I would not be here without it. A special thank you to my father, Sage, for the privilege to work alongside you on this project and over the last three years. Also, thank you to Dr. Hamilton, Dr. Myers, and Dr. McCarty for your shared knowledge and love for all things computer science.

Table of Contents

Table of Figures

Overview

This project seeks to assist the customer relations management (CRM) team at Northwest Nazarene University in the redesign of the Applicant Status Portal. This Portal is a part of a prospective student's portal managed by Slate, the CRM system used and managed by Sage Mwiinga, Joy Wilson, and other members of the CRM team. The goal of this project was to add design elements to make the portal stand out while also being similar in concept to the main NNU website. Additionally, the styling edits will allow for easy maintenance in years to come.

Background

Exposure and familiarity with Slate is something that I have had the opportunity to acquire through my time as an undergraduate at NNU while working for the admissions department and marketing-communications department. Originally starting with logic testing to make sure that all anticipated portal elements are present for prospective students. Progression was made to involve the work of making queries, revising email campaign templates, data entry, and performing data validation. While working with the marketing department, I have had the opportunity to further make edits and adjustments to email templates and campaigns that are regulated through Slate that go out to prospective students.

While these skills and tasks were easy to understand, learn, and complete in the moment, an understanding of what was conceptually and practically being done was developed when those tasks were juxtaposed with a few of the classes I have taken over the last few years at NNU. The classes that aided in the completion of these tasks and ultimately with this project were the Database Design, Big Data Management, and the Introduction to Web Development courses. The concepts and theories of these classes made them some of my favorite courses taken throughout my undergraduate journey.

The database design course gave me exposure to a basis for building queries and the logic rules involved in it. This was put into practice during the course by building the back end for a mass emailer. The emailer application used various tables that held data about the desired email, the recipients, and the attachments the email might hold to produce a campaign that would be sent out. Some of the skills from the emailer that applied to this project were knowledge of joins and the order in which queries are processed.

Big data management allowed for similar learning opportunities, but primarily for the further development of knowledge already learned in database design and how it can be applied to large amounts of data. Although the NNU instance of Slate does not currently warrant the title of being “big data”, its design and intended application allow for it to become a data warehouse within the next couple of years.

Because of the scope of this project, Dr. McCarty’s web development course has shown itself the most useful for this project. This course went over the basic web development languages of HTML, CSS, JavaScript, and PHP. Working in NNU’s marketing- communications Department has shown how all these languages are used together in content management systems such as Drupal and PyroCMS, just to name a few. These management systems allow for editing of websites by multiple users in a way that allows for easy management of the site, user access, and integration between a website and a database of web content. This would be done through database calls are from web pages and content blocks back to a central database.

The Applicant Status Portal is used for students to see the status of their application based on what material is needed to complete their application. The Portal also displays various blocks of content for incoming students that they may need to know such as how to change their major, contact information for academic or financial aid advisors, and what the expected first-semester schedule will look like. Nevertheless, the application status portal hasn't seen many updates or optimization since its creation in 2019 but has been well designed and structured to allow for optimizations when the time comes. One optimization came with Slate's introduction of dynamic block editing and allowance for the use of CSS classes that can be localized to the portal as opposed to individual blocks of content.

Terms

As was mentioned in the background, the areas of computer science this project focused on were database design and web development. Both fields have deeper concepts that can be further explored and developed, which can ultimately lead to careers such as being a webmaster, or database administrator.

Concerning web-related terms, the languages used, and their applications are a good place to start. HTML is the abbreviation for Hypertext Markup Language and is the backbone for most web-related content. It supplies all data on a web page, and the order in which content is rendered. A Cascading Style Sheet, CSS for short, works in conjunction with HTML and allows for the formatting of the page content. With CSS, the color, font, and position of content can be customized to fit the culture of the page. PHP, Hypertext Preprocessor, is a scripting language that is popular in web development for database calls to a server. It allows for a website or page of content to dynamically load content for a page such as different events based on the time a page is accessed. In the context of this project, calls are being made to check for the existence of data in addition to getting information relating to a user.

Content Management Systems are what tie these languages into an easy to work with development environment. These management systems work to allow for multiple users to contribute to the development of a website while helping to keep a uniform look across multiple pages. From a developer standpoint, it makes maintaining and updating large sites less stressful.

Slate is a Customer Relationship Management (CRM) system designed and built by Technolutions to assist in the admissions process efforts of higher education institutions. At NNU, Slate is currently used by the undergraduate and graduate admissions teams to provide prospective students with an avenue to communicate with the admissions team about their application's progress and find details about what is needed to advance their application. Additionally, Slate is used for funneling students through a pipeline into their intended major so they can start conversations with their academic advisors. The job of the CRM system is to make sure the customers, in this case, incoming students, are given a unique experience and interaction throughout their application process.

When it comes to working with any of Slate's portals, and in particular the application status portal, a couple more terms that would be helpful to be familiar with deal with how the portal is created. These terms center around what are described as methods. Methods are the bridge between the queries that a status portal is built on and the different views a portal may have. Views are the different ways a portal can appear and be interacted with by a user. This could be the reordering of the portal's content to load a particular element at the top as opposed to a different placement. But it could also appear as a total redesign of the portal with different elements and blocks. Each method can be composed of multiple queries, but a single method is linked to a single view or type of displayed portal.

Implementation

In preparation to set out and complete this task, there were several articles and webinars shared with me by NNU's CRM Manager, Sage Mwiinga, to provide a basis for setting out to work on the portal. One of the webinars entitled "Portals Methods and Queries and Views, Oh My!" taught me about how the methods, views, queries, and portals are tied together to build the portal. (Technolutions, 2019). A couple of the key things to address before beginning are explained in the article Technolutions published as an introduction to Slate-based portals. The article explains what portals are within Slate and their possible uses. Within Slate, possible uses range from portals that alumni can access to stay connected to a university, another that coaches can use to communicate and stay aware of possible recruits, and status portals for informing users of where they are at in the process of being admitted to a university. Another thing this article is helpful with is proposing questions to the portal developer to get them thinking in the right direction of building a worthwhile, functional, and effective portal. These questions address the type and functionality of the portal, as well as the visual structure of the portal itself.

Portal Functionality

Concerning the types of portals, Slate advertises the option for three types; the Volunteer portal which allows "[…] alumni and other volunteers […] to update their information, sign up to help with events, or submit interview report forms," (Portals 1).

Athletic portals allow for coaches to interact with and manage recruitment for sports. The last type of portal, which is the reason for this project, is the applicant status portal. This type of portal incorporates needed content for and about applying students into a status portal through conditional logic filters. “Conditional logic filters are used to conditionally show fields & prompts based on previously captured data,” (Form Conditional Logic, 1). Conditional logic operates in the same way that logic statements in other programming languages do, checking to see if a statement is true or false based on previously defined and passed data. One example of this can be seen in the application status portal the condition to show an alert about NNU’s Student Health Insurance Plan if a student hasn’t submitted a request to not be enrolled in the plan. There was not much consideration for changing the type of portal since the portal already exists and is used by admitted and prospective students.

The question of functionality asks, what will be possible for the student through the portal? The accepted view of the portal has many functions, but some of the more prevalent functions were the ability to complete or view the FAFSA, verify or change a declared major and entry term, file for the Student Health Insurance Plan (SHIP) waiver, sign up for technology orientation, see their class schedule, visualize a checklist of missing documents needed for admittance, and find contact information for a student’s admissions, academic, and financial aid advisors. While all these functions were kept in the final product, some other functions that were considered were the adding of a social media widget to display either the Northwest Nazarene Facebook or Twitter pages, and the current weather on campus which would provide an upfront way to stay alert of events happening on and around NNU.

The last question to think through in getting ready to tackle this project is how many views the portal should have. In other words, what is the visual experience a user will have while on the portal? There already exists conditionally shown elements in the admittance checklist and additional information blocks of the portal but thinking of how these can be enhanced was considered.

For closed applications, the only functionality that is needed is for the contact information of the assigned admissions counselor and how to reapply. The functionality of this view doesn't need to be reconsidered because there isn't much else that a readmitting student needs to reopen their application.

Development

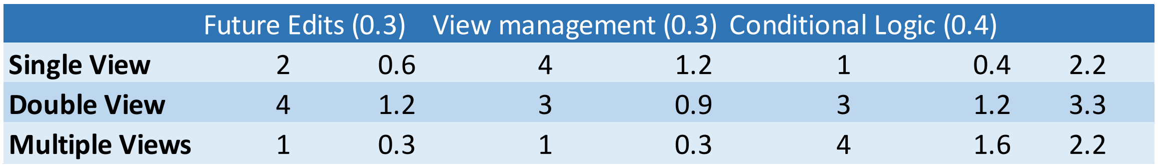
When it came to getting ready to begin development, the first thing that needed to be completed was the creation of a portal to work with. Much of the metadata related to the portal is managed at this phase, which is where custom CSS can be added for that portal. This is also where a default view can be defined for the portal.

The first component of the portal that needed to be recreated was the queries used to pull applicant data, counselor data, and student data for those who have reopened their application. This task within the project required determining joins, relationships between tables, that had to be made, and the correct fields that needed to be pulled to properly build the portal and renaming the fields as needed for the portal. Queries that were used for this project held data relating to applicants, admissions counselors, and readmitted individuals.

The relationship between the queries and methods, as I found it through research and development, resembles a many-to-one relationship. This means that there could exist multiple queries that are associated with a method, and subsequently that any method can have any number of queries that compose it. The resulting methods were one for admitted applications, another for closed applications, and a final one for students who had reopened their applications. Of these methods, the admitted applications method used queried data from Applicant and Counselors queries. Once the queries were built, the methods to run the queries had to be developed. The methods created were related to the created views. Similar to how the relationship between the queries and methods are many-to-one, the relationship between methods and views is also many-to-one.

Two views were created for this portal. The first of which is for those who have been accepted and are in the process of being admitted. This was labeled the “Accepted View”. This is where the bulk of data is displayed and where students have most of their interaction before being admitted. The latter view of this portal is the “Closed View”. This view is for students that have closed their application for the university for any number of reasons but provides the means to reopen it if they choose to do so. One of the questions that had to be addressed when building the views was concerning the number of views to build. Before this project, the two above-mentioned views were the only views that the portal had. Figure 1 shows the decision matrix that was used to help evaluate it was the best option to continue with two views when compared to a single view with a significant amount of conditional logic used throughout the portal and multiple views for many of the possible statuses that a student may have. The matrix shows the comparison between a single view, double view, and multiple views. The single view would be just that would provide a single place to manage edits for the portal but would require a large amount of conditional logic. Having multiple views seeks to eliminate the need for this logic by building queries and methods to fit all the possible statuses an applying student might have.

Maintaining a portal with this type of structure could be interesting in concept, but in practicality would pose to be tedious to perform edits on and in the initial setup to consider all possible scenarios.



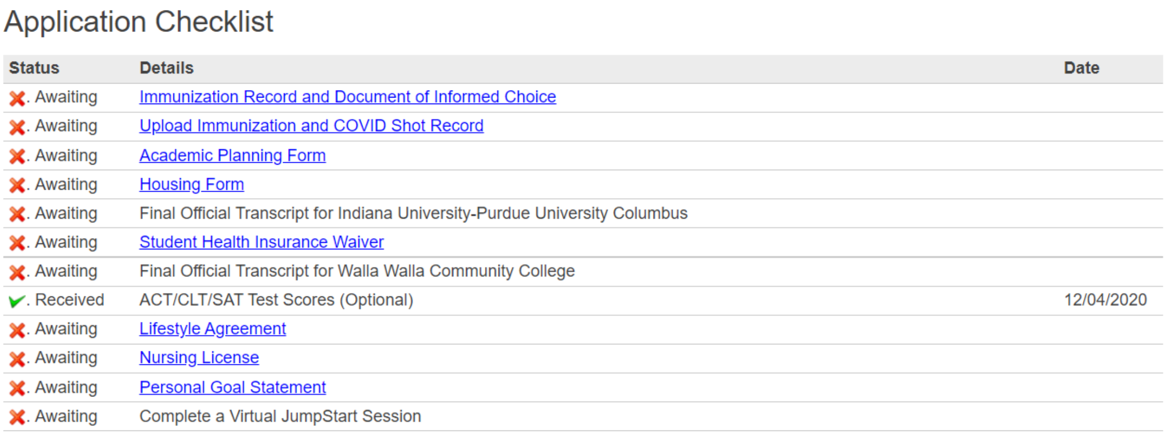
	Future Edits (0.3)		View management (0.3)		Conditional Logic (0.4)		
Single View	2	0.6	4	1.2	1	0.4	2.2
Double View	4	1.2	3	0.9	3	1.2	3.3
Multiple Views	1	0.3	1	0.3	4	1.6	2.2

Figure 1: Decision Matrix Comparing the Number of Views

Notes. This matrix compares three solutions based on the importance of different evaluation metrics. For this project, the need for conditional logic and view management were compared along side of anticipated ease of future edits. By using the matrix, continuing to use two views showed to be the solution with the highest score, and would be the best option to continue with.

This completed the recreation of the portal for making edits and would allow for a sandbox to experiment and freely create with CSS. Creating a styling system to use is best done using classes. These are easy ways to replicate a certain style across multiple elements of a web page in an easy-to-maintain system. This would also help in simplifying some of the static blocks that had predetermined styling within them. One of the issues that had to be resolved was overriding styling that was enforced on certain blocks of the portal. In the CSS world, there is a way "[…] to override a style that cannot be overridden in any other way," (CSS 1). This is through the `!important` rule. This rule forces a particular style to show no matter what any localized styles are. A couple of the applications, which was capitalized on for this project, concerned a checklist table that shows students the status of documents needed for their admittance, and the formatting of a block that shows admissions counselor contact information.

Before this project, the checklist table had no styling attached to it and was styled by Slate’s default styles. Figure 2 shows the table in its prior state with a simple design that showed no character. The rows of the table were all white and had minimal spacing between them, something that make use of the checklist more tedious to navigate. The links to other pages also showed up in a blue that doesn’t match or complement any of the university colors. Using the `!important` rule to override and force styles allowed for what is seen in Figure 3. The new table introduces accented colors in the header and font color, that match NNU brand of red, to replace the typical blue that often appears when notating a hyperlink. Additionally, a visual feature that informs the user of the current position on the table provides a small amount of feedback to the user.

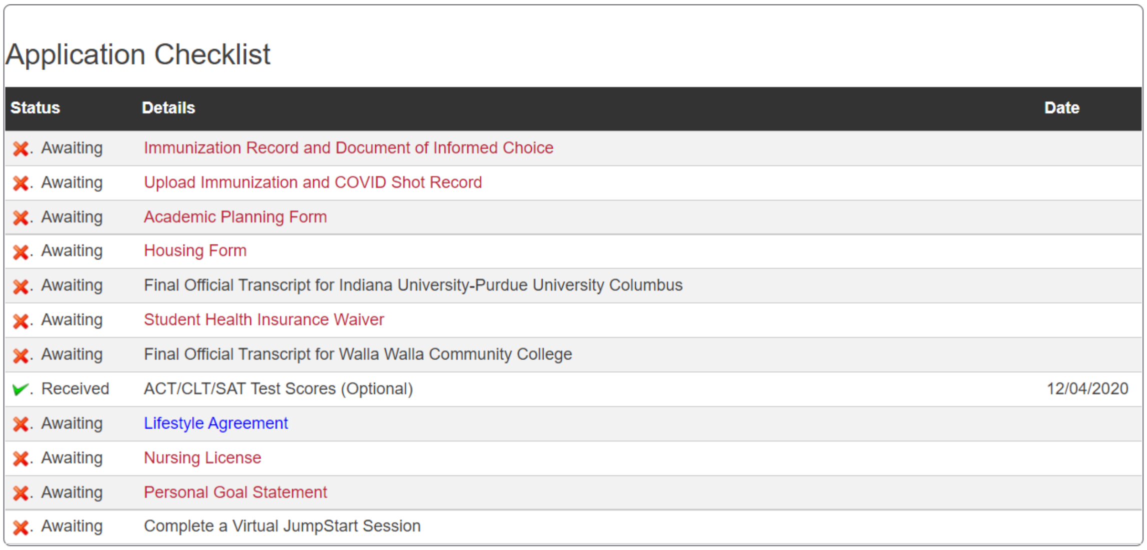


Application Checklist

Status	Details	Date
✗. Awaiting	Immunization Record and Document of Informed Choice	
✗. Awaiting	Upload Immunization and COVID Shot Record	
✗. Awaiting	Academic Planning Form	
✗. Awaiting	Housing Form	
✗. Awaiting	Final Official Transcript for Indiana University-Purdue University Columbus	
✗. Awaiting	Student Health Insurance Waiver	
✗. Awaiting	Final Official Transcript for Walla Walla Community College	
✓. Received	ACT/CLT/SAT Test Scores (Optional)	12/04/2020
✗. Awaiting	Lifestyle Agreement	
✗. Awaiting	Nursing License	
✗. Awaiting	Personal Goal Statement	
✗. Awaiting	Complete a Virtual JumpStart Session	

Figure 2: Original Checklist Style

Note. This figure is an example of what a current student would see on their portal.



Application Checklist

Status	Details	Date
✗. Awaiting	Immunization Record and Document of Informed Choice	
✗. Awaiting	Upload Immunization and COVID Shot Record	
✗. Awaiting	Academic Planning Form	
✗. Awaiting	Housing Form	
✗. Awaiting	Final Official Transcript for Indiana University-Purdue University Columbus	
✗. Awaiting	Student Health Insurance Waiver	
✗. Awaiting	Final Official Transcript for Walla Walla Community College	
✓. Received	ACT/CLT/SAT Test Scores (Optional)	12/04/2020
✗. Awaiting	Lifestyle Agreement	
✗. Awaiting	Nursing License	
✗. Awaiting	Personal Goal Statement	
✗. Awaiting	Complete a Virtual JumpStart Session	

Figure 3: Resulting Checklist Styles

Note. This is the updated version of the Checklist that prospective students will see.

Conclusion

The results of this project were successful on two fronts. Firstly, the simplification of the styling used on the portal into an easy to maintain and manage the system. A total of 15 classes were introduced to standardize the look, fonts, and colors that appear throughout the portal. The classes allowed for overriding predefined styles so that the portal's web culture reflects what is seen on the main NNU website using brand colors. Lastly, a social media widget, to keep applicants up to date with events going on and around campus was added which will provide students with the means to stay informed and aware of events and news that happens on and around the NNU community.

In addition to this, CSS was added to create a unique yet familiar portal when compared to the university website. Future work that can be done with this portal would be to explore the implication of CSS classes that are suited for mobile and desktop environments. One example is tailwind CSS as this is already being implemented in the

NNU website. The edits for this portal will be implemented for the undergraduate and graduate classes of Fall 2022. Students will be presented with relevant information about their application and their future at NNU, drawing them into being Here for Good.

References

CSS the !important rule. CSS !important Property. (n.d.). Retrieved November 10, 2021, from https://www.w3schools.com/css/css_important.asp.

Form Conditional Logic. Slate Knowledge Base. (n.d). Retrieved November 10, 2021, from https://knowledge.technolutions.com/hc/en-us/articles/360033289752-Form-Conditional-Logic.

Portals – Slate Knowledge Base - knowledge.technolutions.com. knowledge.technolutions.com. (n.d.). Retrieved November 9, 2021, from https://knowledge.technolutions.com/hc/en-us/articles/360033470831-Portals.

Technolutions. (2019). *Portals Webinar Series (Part 1 of 3)*. *Youtube*. Retrieved August 4, 2020, from https://www.youtube.com/watch?v=FOsojOEmExw.