Butler Theatre Database Project

Justin Rice, Andrew Nesler, Steven Nirenberg, Gwen Spencer, and Grace Maynard

EPICS: Engineering Projects in Community Service
Spring 2020

Acknowledgements

A big thank you to Nate Partenheimer for his endless assistance in developing our website and database. His continuous support throughout the lifetime of this project has been truly invaluable. Thank you to Megan Wiegand for allowing the EPICS class to work on her project. We have gained a significant amount of knowledge from working on this project and are thankful that the theatre department turned to EPICS for this project. Many thanks to Dr. Panos Linos for guiding us throughout this project and for encouraging us in completing this project.

Table of Contents

Glossary and Terminology

Adobe XD: A wireframe design platform for website development. Adobe XD allows users to create a click-through prototype of the wireframe design and easily share designs with team members.

ERD: An entity relationship diagram (ERD) describes the structure of a database. An ERD shows the relationships between tables and attributes of the data within those tables.

Javascript: JS is a programming language often used in many web applications and interacts with CSS & HTML languages.

mySQL: MySQL is an open-source relational database management system.

PHP: A popular general-purpose scripting language that is especially suited to web development.

Rest API: An API is an application programming interface. It is a set of rules that allow programs to talk to each other. The developer creates the API on the server and allows the client to talk to it. Representational state transfer (Rest) is a software architectural style that defines a set of constraints to be used for creating Web services.

UUID: A universally unique identifier is used to ensure that each entry in the database has a unique identifying key.

WordPress: An open-source software written in PHP used for designing blogs, websites, and apps. WordPress has many plugins available for easy website customization, as well as pre-designed website templates. Websites can also be customized with PHP, HTML, or CSS.

CSS (Cascading Style Sheets): A style language used in this context for website formatting. CSS is optimal to use to describe the fonts, colors, and other design features a website should use.

Abstract

The Butler University Theatre Department houses over 15,000 garments that are used both by the university and outside vendors. Due to the large number of garments, there is currently no efficient method for handling and organizing these garments. This project seeks to develop both a database and a web application to improve the efficiency of the Theatre Department in locating garments and for outside users in renting the garments. The database will house all of the applicable information about the garments owned by the department and the web application will be an online store that will allow users to search and filter for specific garments for the purpose of renting them. For the Spring of 2020, the Butler Theatre EPICS team was able to develop the framework for the database and begin the front-end development of the website.

Chapter 1: Introduction

Section 1.1: Problem Statement and Objectives

The theatre department has well over 15,000 articles of clothing in their collection. Organizing and categorizing these costumes is a monumental task and having a system to keep track of the different pieces would make the lives of the staff much easier. In figure 1 shown below, the storage area where all of the garments and costumes are housed can be seen. The main goal of this semester was to design and develop a database system and application for the Butler Theatre Department to help organize their costumes and to increase the efficiency of accessing items. There are several objectives that were identified at the beginning of the semester that were necessary to achieve the goal of a working website and database. The first objective was to identify a suitable website that could both complement the database and offer e-commerce capabilities. The second objective was to design and build a database that could store all of the data associated with each garments specific attributes (i.e. color, size, era, sex, etc.) and organize in a way that was convenient for the client's needs. The last objective was to identify and begin to build a bridge between the front-end (website) and back-end (database) through developing a custom API layer.

Figure 1: Racks holding garments in the storage and scene shop

Section 1.2: Motivation and Rationale

The current system for renting garments does not have a web interface or any automated system. As a result, it is difficult for users to know what the garments look like that they are renting and the condition of these garments. By creating a database and web application, users will be able to see photos of the garments remotely prior to renting and will be able to select garments that fit their exact needs. This system will also provide a way for the Theatre Department to easily track orders and to know which garments are currently being used. Furthermore, a web application will be more accessible to users, because it can be easily accessed anywhere and can be set up for use by screen readers and other accessibility tools. This will ensure that any individual looking to rent garments from our Theatre Department will have no obstacles in doing so.

Section 1.3: Description of the Customer

Renowned as one of the best programs in the country, Butler Theatre strives to push the boundaries of our art as we create innovative productions and instill in our graduates an entrepreneurial spirit that will help them to succeed in an ever-changing market. With a strong liberal arts foundation, we foster the development of well-rounded, critical-thinking artists who can plot their own successful path in theatre and in life.

Section 1.4: Approach to the Project

The approach to this semester's project was to establish a dedicated virtual machine on Butler's infrastructure, and create a mySQL relational database for garment attribute information storage. From there we downloaded the WordPress package to begin development on a front-end interface. We also used Adobe XD to prototype a wireframe design to show the client, providing a foundation for the front-end interface design and expectations moving forward. Once the garment database and WordPress site were created, we identified the approach to development of a custom-tailored API layer to bridge the interface to the database storage.

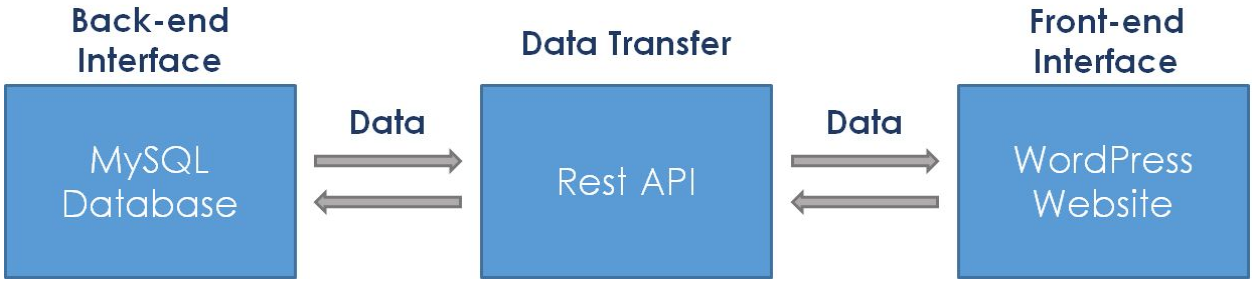

Figure 2: Diagram of how the data will flow from the database to the website. The website will pull attributes, images, etc. from the database and the API is the facilitator.

Chapter 2: Requirements Specifications

Section 2.1: Project Requirements

The client had many goals for the appearance and functionality of the website. Primarily, the client wanted to conclusively store inventory information, as well as offer the ability for visitors to rent and view inventory remotely. As a result, the website will need to include shopping interfaces and form submission, ability to filter items, view images of garments, and search for a specific piece by attribute. The client also wanted the ability to show the availability of an item, how it should be washed, the current condition, and descriptionary attributes (style, color, decade). Another, rather laborious process of this project will be adhering barcodes to garments with serial numbers referencing UUID in the database to track items. This will allow for potential development of administrative interfaces to control garment inventory and confirm stock, later down the road. This project has no specific timeline, since the current system for the Theatre Department is still fully functional. Maps from the client showing the layout of the warehouse can be found in the Appendix in Figures A1 and A2.

Chapter 3: Architecture

Section 3.1: Overview of System Architecture

The current project and all its associated packages and processes are hosted on a virtual machine on Butler's infrastructure. This will ensure long-term maintenance and support, as well as simplify the porting of the network interface to a sub-domain of the University.

Within the virtual machine we used Vagrant to deploy a functional instance of a standard WordPress application. We decided to begin implementation of a web interface using WordPress (WP) due to the simplicity and wide customizability of the platform. Rather than building our own LAMP stack and writing custom code and outputting HTML directly, we decided that the group would accomplish the most together through a simplified UI. From the fresh WP instance, we installed Advanced Custom Fields (ACF) to allow more flexibility through a friendly development user interface. While there exist methods for directly implementing an existing wireframe design such as the one we created to a wordpress website, due to our current understanding we decided to continue a wordpress-only implementation to see how far that got us. We started by creating custom fields and post types to suit our needs for this project. ACF allowed us to create different forms of content on the website beyond posts and pages. This is essential to begin development of a highly specific application in wordpress, since the original function of the service is to simplify creating websites for blogs or that fanfiction that you and your internet friends in middle school wrote. For example, we have created a "shirts" post type, so each garment that is classified as a shirt will be able to be displayed similarly on the finished

website. Refinement and organization of the implementation of ACF into WP is not yet fully understood, so much of this portion of the development could still be up for discussion.

WordPress offers many other advantages, including already existing implementation through mySQL relational databases. Given this convenient setup, we went into the VM, set correct permissions and created an additional database specific to garment attribute data storage. To avoid full command line implementation, we installed phpMyAdmin for a database management UI. Queries for creating the databases and mapping primary keys were created in shared text files, adapted as we learned more about our clients needs, and eventually injected into the phpMyAdmin interface to create the ERD in Figure 2.

Our first approach to tackling the garment storage was to create a mySQL relational database, alongside the wordpress db, called garment_inventory. This primary table aims to store general attribute information, with lookup tables attached to simplify sorting later. Additionally, each garment type has a table for their own specific qualities. This current design has been implemented through query scripts mentioned prior and saved to a common github repository, which simplifies further changes and design alterations down the road. Additionally, improvements to this current design have been discussed in detail with relevant professionals and will be mentioned in the Future Work section. Database design follows the general format seen in Figure 2.

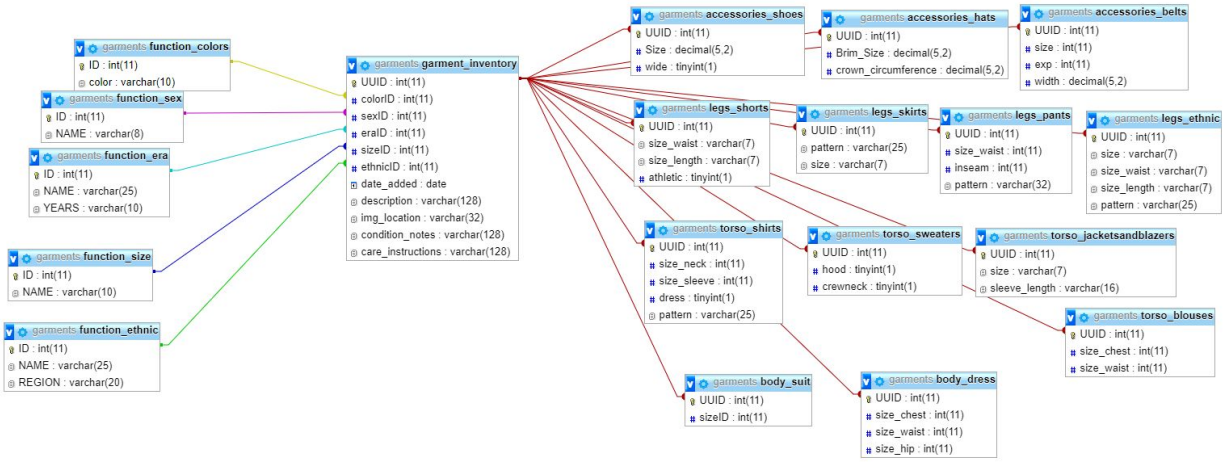

Figure 3: Database ERD

Functional tables on the left will hold generic standardized information such as color, size, gender, era and ethnicity. This naming and structure is pseudonymous, since these will eventually be adapted to the format of lookup tables, but awaits implementation due to time restrictions. The database will feature one primary table on the right will be used to store a UUID, description, timestamp and other universal data, while the tables on the left will store garment-type specific qualities. There has been much debate regarding the potential hurdles that

will occur down the road stemming from this choice in database design, so development has remained general until this time. We instead focused on gathering the clients needs and identified attributes that needed included. This current structure will require significantly more complex queries when developing the API layer. Due to the desire to create a rental interface, document store database systems such as MongoDB may prove more efficient later as well. Another potential fix would be to use a database supporting JSON data types. This would allow us to use PostgreSQL to store garment specific data in a JSON map that would require a single column. This JSON map can then be queried for any of its attributes or even indexed itself. Implementing such a system would improve search efficiency as well as simplify development of the API.

```
CREATE TABLE garment_inventory(
    UUID INT,
    IDENTITY(1,1) NOT NULL AS UUID, -- how to do pls
    colorlookup VARCHAR(16) NOT NULL,
    genderlookup INT NOT NULL,
    eralookup INT,
    sizelookup INT,
    ethnicity int,
    date_added DATETIME not null,
    description varchar(128),
    img_location varchar(32),
    condition_notes varchar(128),
    care_instructions varchar(128),
    PRIMARY KEY (UUID),
    FOREIGN KEY (ethnicity) REFERENCES lookup_ethnic(name),
    FOREIGN KEY (sizelookup) REFERENCES lookup_size(name),
    FOREIGN KEY (colorlookup) REFERENCES lookup_colors(name),
    FOREIGN KEY (genderlookup) REFERENCES lookup_gender(`gender`),
    FOREIGN KEY (eralookup) REFERENCES lookup_era(name)
);
```

Figure 4: Sample create statement for primary table garment_inventory

Figure 4 depicts a sample of the create query for our primary garment inventory table. As commented, the first integer value is not properly configured to increment a unique identifier. The goal for that column would be to create a field that represents the garment universally, and embed a function to automatically increment as data fields are added, ensuring no duplicate values as well. Some functions such as this can be implemented through the user interface of PHPmyAdmin, however cannot be executed without understanding of the function itself. Because of this, we have opted to create the database in a text file, making changes based on the needs of the UI when necessary.

For our purposes, we will need to construct an additional software layer that will query specified data from the garment database, and hand it off to the wordpress application to process and output in a friendly manner. Nate gave us significant insight as well as a brief lesson in Python API creation using our VM instance, but due to the preferences of those writing the code, we

decided to stick with javascript as our language instead. This API layer will translate buttons in search functions and site navigation, and translate them into queries familiar to a SQL database. For example, if you navigate to search for all blue hats size 7.5 , the database needs that in its own language. So the API layer will attach the fields sort by mens, blue, pants, into a proper pseudo-query:

SELECT * FROM garment_inventory WHERE hat_size = 7.5 && color = blue

An API layer accomplishes this, by using lookup tables to map variables to form data with sql statements, the front end interface will be able to read from the relational storage. Further notes regarding specifics will be provided with the project files.

Section 3.2: System Platforms

System is currently hosted on a virtual machine running on Butler's infrastructure for a development environment. This development environment encloses all the necessary packages for Wordpress and mySQL to host the program. Development should continue in this secure environment until published, when default passwords must be changed and back-end management tools secured. Database management accessible through PHPmyAdmin webUI portal 10.131.3.197/phpmyadmin/. Wordpress management provided from same.host/wp-admin/. Authentication information will be provided securely with project files.

Chapter 4: Design

Section 4.1: User Interface Overview

We used AdobeXD in order to design a wireframe of the user interface. This process helped us determine how we would like users to interact with the website. We also received input from our client on our design in order to understand how they would want users to be able to interact with the website. Some pages of the wireframe are included below in Figures 5 and 6. There are more images of the design work in the Figure A3 and A4 in the Appendix. The wireframe may be viewed in its entirety at the following website:

https://xd.adobe.com/view/5508ef11-f71a-4a69-7eb2-3ca67ed7ac43-2339/.

For the design of the wireframe, a similar style to Butler's main website (Butler University) was chosen. A color palette that has similar colors to Butler's brand was used throughout the wireframe and the recommended substitute fonts, Georgia and Helvetica, were used primarily. The goal of the design of the website was to showcase a modern and simplistic design, but to

also encourage users to rent from the garments available. We consulted Target.com (Target) for inspiration for an e-commerce website.

To help display all the garments that the Theatre Department has to offer, professional stock photos of the garment warehouse were acquired from our client. These photos have been used both on our wireframe and on our WordPress site.

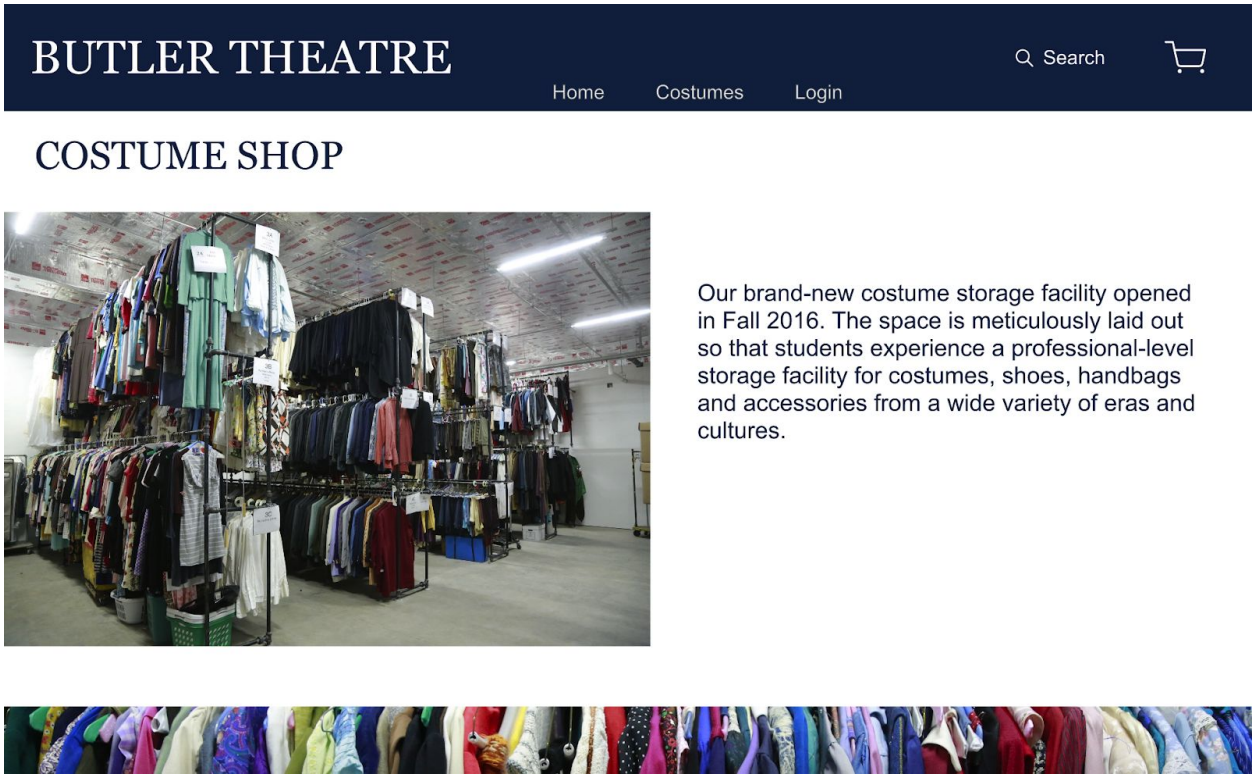

Figure 5: Home page of the wireframe design

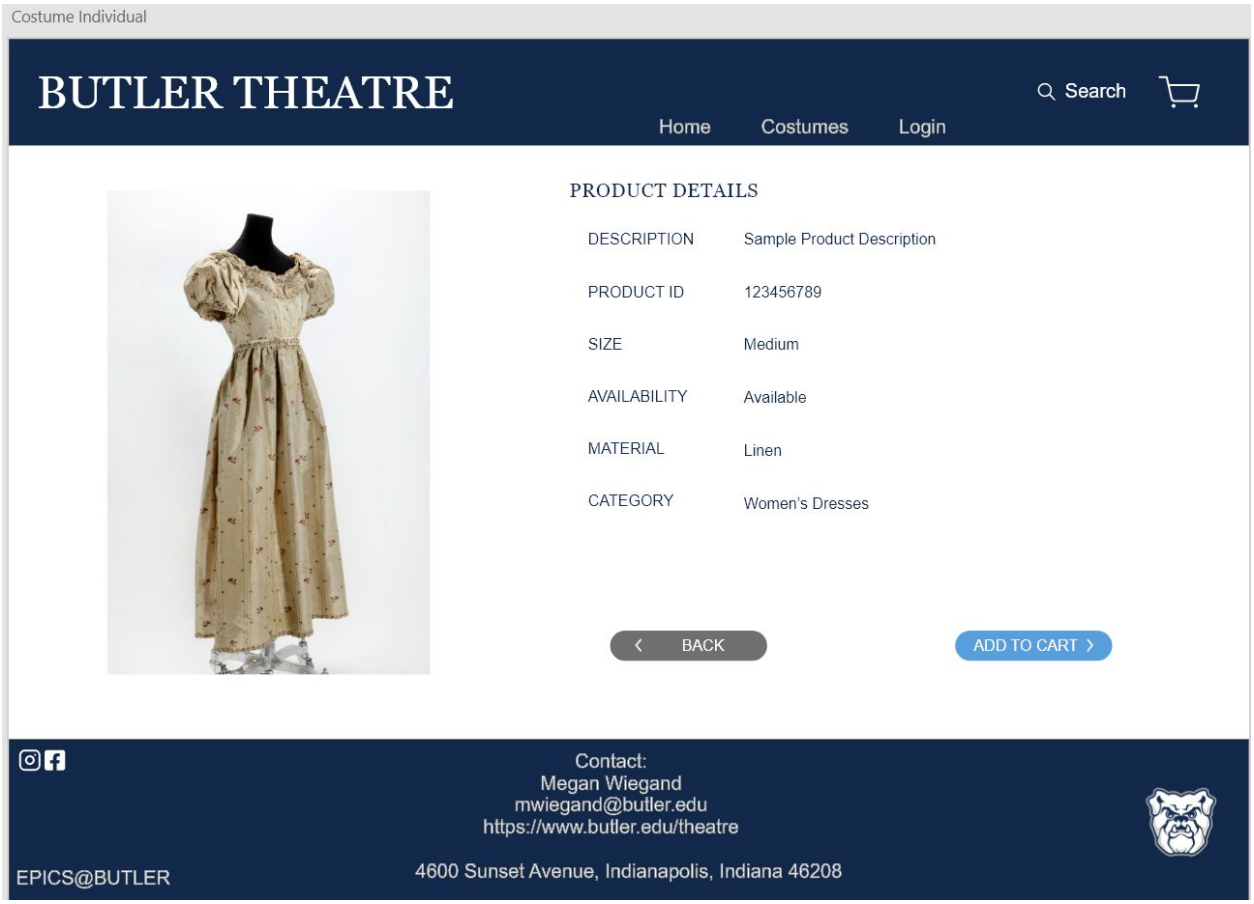

Figure 6: Product Description Page

After having the design of the wireframe approved by the client, we began designing a WordPress website based on the wireframe. As shown in Figure 7, we successfully began implementing key features of the wireframe into WordPress. We imported images from the wireframe onto the website, established matching fonts and color schemes, and developed a similar menu bar. As further discussed in Chapter 8, future groups will need to continue developing the WordPress website to match the design of the wireframe.

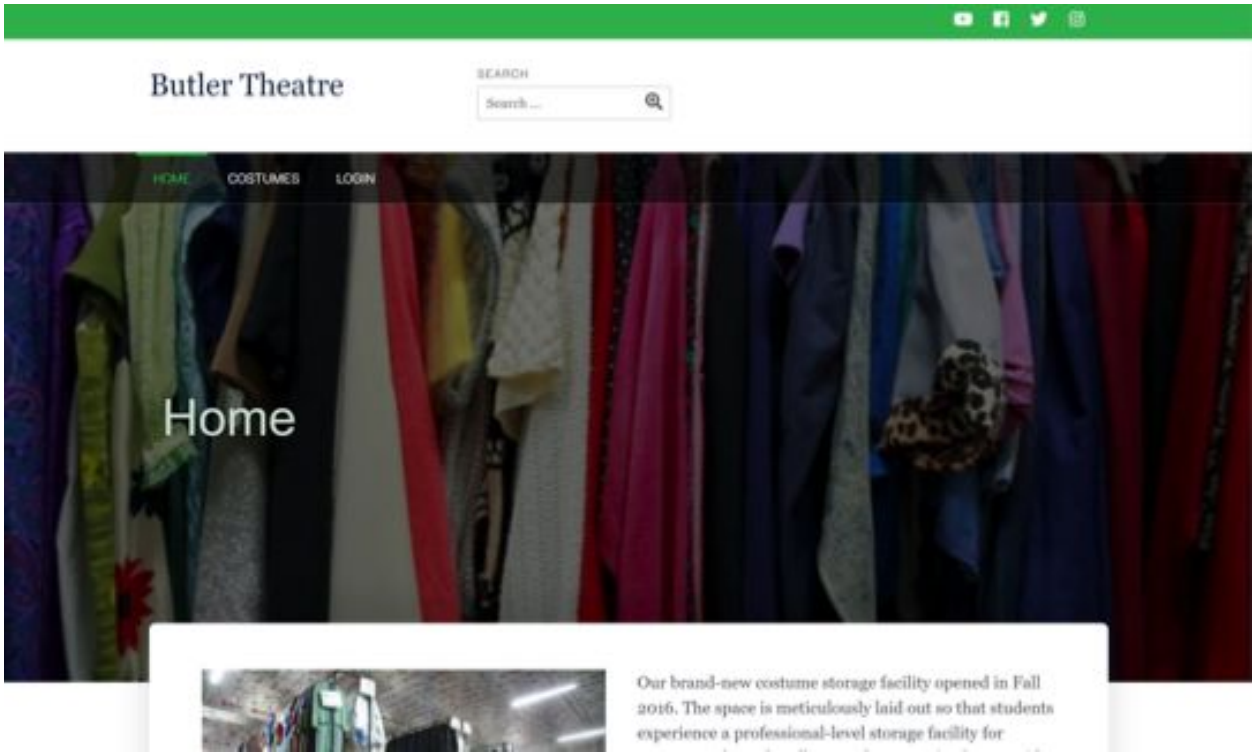

Figure 7: WordPress Homepage

Chapter 5: Implementation

Section 5.1: Implementation Languages

There were several languages used for different aspects of the project. For the WordPress package, PHP, CSS, and mySQL were required for installation and were used in a virtual machine running with vagrant and virtualbox. To get us started on the API layer, we introduced some essential ideas and functions using Python in a virtual machine. For the future the API will be written in Javascript using Express.js and Node.js. For the database

Section 5.2: Distribution of Work

The group was split up into two different sections: front-end application and the database and back-end application. Gwen, Grace and Justin were primarily responsible for the design and implementation of the front-end work. They designed the wireframe in Adobe XD and began the implementation of the design into WordPress. Andrew and Steven were primarily responsible for all of the back-end work. Andrew designed and tested the create queries for the database. Steven was responsible for the research into the API. Their combined efforts, with input from Nate

Partenheimer, Michael Burroughs, and Butler's IT department worked to create a functioning relational database in the development environment.

Chapter 6: Quality Assurance & Testing

Due to several of the unforeseen challenges presented in this project, there is not currently a conclusive working prototype of the entire system. The website does host a regular web page with static data displayed and the database is working and capable of storing data. However, due to time constraints a working API was not developed and implemented.

Chapter 7: Project Organization & Management

Section 7.1: Team Organizational Structure

The team is structured such that every individual is responsible for their own tasks. The assigned roles with the team include: team leader, client liaison, database developer(s), and documentation and website lead. The team leader is the head member of the group and coordinates all group assignments.

Section 7.2: Team Roles and Contributions:

Team Leader: Justin Rice

Justin is a senior from Fowler, IN studying economics and mechanical engineering with a minor in computer science. He was the team leader and helped guide the team towards achieving the stated goals. His primary responsibilities were assigning tasks, writing weekly summary reports, and communicating with Dr. Linos. He also helped to design and create the website wireframe in Adobe XD and set up the WordPress site.

Client Liaison: Gwen Spencer

Gwen is a senior from Tampa, FL majoring in actuarial science with a minor in business administration. She was the primary source of communication between the Theatre Team and the product owner. Her main contributions were in the front-end side of the project. She helped to create the wireframe for the Theatre website and to set up the WordPress site.

Database Developer: Andrew Nesler

Andrew is a senior from Chesterfield, Missouri studying computer science. He maintained communications and worked regularly with Nate Partenheimer to progress implementation on the database infrastructure. Together, they set up the working environment for website and

database administration. Andrew then focused on database development and created a query-generated ERD reflecting the attributes from the needs of the client. To do so he worked with the client to discuss how the database worked for her and what data is most relevant to her needs. Andrew will continue progress on this project after the semester and into the fall, since full scale pictures of a project can be difficult to communicate.

Database Developer: Steven Nirenberg

Steven is a sophomore from Coconut Creek, FL, majoring in software engineering. His major contributions were in the back-end side of this project, setting up the restAPI for the database and for the website.

Documentation and Website Lead: Grace Maynard

Grace is a junior from Normal, IL majoring in math with a minor in management information systems. She created and updated the Theatre Team website, documented the progress made as a team, and kept track of any essential documents. She helped construct the wireframe for the Theatre website and build the WordPress site, so her main contributions were seen in the front-end development.

Section 7.3: Project Management Process

The team used several methods to manage and achieve the tasks that had been set. A group chat and email were the primary forms of communication between the team and Microsoft Planner was used for accountability and to set deadlines for the different aspects of the project. Google Drive was used to organize our presentations and any word documents. Microsoft Planner was used to help keep track of documents. The EPICS website was utilized as a repository for our Sprints and for each WSR. Lastly, Github was used to store all of our code.

Section 7.4: Weekly Status Reports

A Weekly Status Report was shared at the beginning of each week to address red flags, issues, accomplishments, and action items for that week. They allowed our team to reflect on the progress we had made, as well as determine the tasks on which we needed to focus. Each WSR is attached below.

WSR 1:

WEEKLY STATUS REPORT (WSR)
February 10, 2020

TO: Grace Maynard, Gwen Spencer, Steven Nirenberg, Andrew Nesler, & Dr. Panos Linos.
FROM: Justin Rice
SUBJECT: Status report for week of February 10th

I. RED FLAGS: Lack of access to the Fall 2019 Butler Theatre groups artifacts and data.

II. ISSUES: Accessing the artifacts and data

III. ACCOMPLISHMENTS (dates):

February 05, 2020
- Completed our Sprint 1 presentation.
- Updated our page on the EPICS website.

February 10, 2020
- Meeting with Megan Wiegand to tour the storage warehouse and discuss the needs and goals for the project.

IV. ACTION ITEMS FOR FOLLOWING WEEK 2/10/20 - 2/15/20:

- Group photo
- Data acquisition
- Determine clients needs and desires
- Identifying starting point for project

WSR 2:

WEEKLY STATUS REPORT (WSR)
February 17, 2020

TO: Grace Maynard, Andrew Nesler, Gwen Spencer, Andrew Nirenberg, & Dr. Linos
FROM: Justin Rice
SUBJECT: Status report for week 2/17 – 2/23

I. RED FLAGS: None at the moment

II. ISSUES: None at the moment

III. ACCOMPLISHMENTS (2/10/20 – 12/17/20):

2/10/2020

- Met with client to discuss goals for project and see the physical location of the costumes.
- Divided into website and database teams.
- Team picture taken.

IV. ACTION ITEMS FOR FOLLOWING WEEK (dates):

- Identify the best database.
- Begin building the database.
- Continue wireframe designs in Adobe XD.
- Identify a website builder to use.
- Remaining on task and target and continuing to make progress.

WSR 3:

WEEKLY STATUS REPORT (WSR)
February 24, 2020

TO: Grace Maynard, Gwen Spencer, Andrew Nesler, Steven Nirenberg, & Dr. Linos
FROM: Justin Rice
SUBJECT: Status report for week 2/24/20 to 3/02/20

I. RED FLAGS: N/A

II. ISSUES: Debugging old database and entering test data

III. ACCOMPLISHMENTS:

2/19/20

- Made good progress on the wireframe design for the website.

2/24/20

- Met with Nate Partenheimer to discuss Fall 2019's database and discuss the best moves forward.

IV. ACTION ITEMS FOR FOLLOWING WEEK (2/24 – 3/02):

- Complete and give Sprint #2 presentation. (All)
- Finish up the wireframe design for website. (Grace, Gwen, & Justin)
- Debug and begin testing old database (Steven & Andrew).

WSR 4:

WEEKLY STATUS REPORT (WSR)
March 2, 2019

TO: Gwen Spencer, Grace Maynard, Steven Nirenberg, Andrew Nesler, and Dr. Linos
FROM: Justin Rice
SUBJECT: Status report for week 3/02 – 3/08

I. RED FLAGS: Getting the WordPress site set up to begin database design and development.

II. ISSUES: Django interface design.

III. ACCOMPLISHMENTS (dates):

2/24
- Met with Nate Partenheimer to discuss issues with database as well as steps to move forward.

2/26
- Gave Sprint #2 presentation.

IV. ACTION ITEMS FOR FOLLOWING WEEK (3/02 – 3/08):

- Develop WordPress website (Gwen, Justin, & Grace)
- Research and develop Django application (Steven & Andrew)

WSR 5:

WEEKLY STATUS REPORT (WSR)
March 30, 2020

TO: Andrew Nesler, Steven Nirenberg, Grace Maynard, Gwen Spencer, & Dr. Panos
FROM: Justin Rice
SUBJECT: Status report for week 3/30/20 – 4/05/20

I. RED FLAGS: N/A

II. ISSUES: Adding post-types and fields to WordPress site. Reconstructing database on Wordpress VM to ease bridging into UI. Requires further database construction and collaboration with client regarding specific garment attributes and datatypes.

III. ACCOMPLISHMENTS (dates):

3/09/20
- Gave Sprint #2 Presentation

3/11/20
- Met with Nate Parthenheimer & Co-worker to go over how to add post-types & fields to WordPress Site.

IV. ACTION ITEMS FOR FOLLOWING WEEK (3/30 – 4/05):

- Create child-theme for WordPress (Grace, Gwen, & Justin)
- Complete ERD diagram and associated code (Steven & Andrew)
- Finish adding post-types & fields (Grace, Gwen, & Justin)

WSR 6:

WEEKLY STATUS REPORT (WSR)
April 6, 2020

TO: Gwen Spencer, Grace Maynard, Steven Nirenberg, Andrew Nesler, & Dr. Linos
FROM: Justin Rice
SUBJECT: Status report for week 4/06/20 – 4/12/20

I. RED FLAGS: N/A

II. ISSUES: Reaching out to Meghan (Client) to discuss some questions that we had for her.

III. ACCOMPLISHMENTS (3/30/20 – 4/05/20):

4/01/20

- Made child theme
- Identified path to meshing our ideas and project

IV. ACTION ITEMS FOR FOLLOWING WEEK (4/06 – 4/12):

- Assigning tasks for poster (Everyone)
- Working on meshing Wordpress & SQL database (Andrew & Steven)
- Building rest API for Wordpress site (Andrew & Steven)
- Editing child theme on WordPress (Gwen, Grace, & Justin)

WSR 7:

WEEKLY STATUS REPORT (WSR)

April 13, 2020

TO: Grace Maynard, Gwen Spencer, Andrew Nesler, Steven Nirenberg, & Dr. Linos
FROM: Justin Rice
SUBJECT: Status report for week 4/13/20 – 4/19/20

I. RED FLAGS: N/A

II. ISSUES: N/A

III. ACCOMPLISHMENTS (4/6 – 4/12):

4/8/20

- Worked with Nate Partenheimer to identify the best route for implementing an API into out system.

IV. ACTION ITEMS FOR FOLLOWING WEEK (4/13 – 4/19):

- Continue working on WordPress theme layout (Gwen & Grace)
- Continue researching API (Andrew & Steven)
- Finish ERD diagram & Database layout (Andrew)
- Work on Sprint 4 Presentation (All)
- Work on the final report & presentation (All)
- Work on project poster (Justin)

WSR 8:

WEEKLY STATUS REPORT (WSR)
April 20, 2020

TO: Gwen Spencer, Grace Maynard, Andrew Nesler, Stephen Nirenberg, & Dr. Linos
FROM: Justin Rice
SUBJECT: Status report for week 4/20 – 4/27

I. RED FLAGS: N/A

II. ISSUES: Working on final report & presentation

III. ACCOMPLISHMENTS (dates):

April 15, 2020
- Sprint #4 Presentation

IV. ACTION ITEMS FOR FOLLOWING WEEK (dates):

- Continue researching the Rest API info and compiling data. (Steven)
- Finalize ERD diagram & database code (Andrew)
- Finish report and presentation (All)
- Finish poster (Justin)
- Contact client with final details and semester update (Gwen)
- Condense all semesters info into one singular location (Grace)
- Update as much of WordPress site as possible (Gwen, Grace, & Justin)

Section 7.5: User's Manual

As discussed previously, the system as a whole is not implemented completely and functions only independently. Therefore, a user manual is not necessary at this stage of the project. In the future, after the API is built and can effectively communicate with the WordPress site and the database, a user's manual will be an important document for the client and other users.

Chapter 8: Future Work

Section 8.1: Future Work

The main work to be completed in future semesters will be connecting the database and the front-end user interface, polishing the front-end design elements, and testing for quality assurance and breakability.

For the front-end user experience, the website will need to be further updated to match the design of the wireframe. We were successful in building a framework for the website this semester and hope that future groups will complete the rest of the website design. Some specific features that will need to be implemented include e-commerce capabilities, accessibility features, login options, and auth0 login for administrative purposes. The website should also be optimized for use on all devices (computers, tablets, phones, etc.). Lastly, the administrative view of the website should be intuitive enough for an individual unfamiliar with the software to use.

Future Groups will be expected to continue to develop and hopefully complete the REST API. As of right now, the languages of choice will be Express.js and Node.js. While changes may occur, this is not expected. A current member, who will return, is learning basic javascript and plans on learning some Node and Express in preparation for EPICS fall 2020 semester.

Also, the aforementioned adaptation of the table to use page views using PostgreSQL has had little research or reflection in current design, even though it will likely be the most efficient strategy to read the database with a rest API.

For the back-end there are several ways that future teams could improve upon the current infrastructure. In the future, groups should also consider how images and data will be inputted into the database. An easy method for uploading image files into the database should be created, as well as an intuitive way to input garment data. As the database was developed, constant reconstruction and re-injecting SQL scripts was necessary to adapt over prior table constraints. Changes such as migrating functional tables to lookup tables using natural keys instead of integer primary keys are halfway done, as well as ensuring those lookup tables are properly associated to the primary garment_inventory table. The implementation of an Identity function to automatically increment a universally unique per-garment identifier has not been completed at the time of this writing, due to time and understanding (or lack thereof) of SQL queries.

Future groups should also work to test the application for quality assurance and breakability. Since our client will be the individual inputting data into the database, it is important that the client would have no opportunity to interfere with the database design. Also, any user could visit the website and search for garments, so the website should be tested to ensure that any user could easily use the search capabilities of the website.

Bibliography

Butler University. (2020). Retrieved February 10, 2020, from https://www.butler.edu/

Butler University 2019 Style Guide. (2019). Retrieved from https://www.butler.edu/sites/default/files/bu-brandguidelines2019-web.pdf

Schäferhoff, N. (2016, January 21). How To Create And Customize A WordPress Child Theme. Retrieved March 2020, from https://www.smashingmagazine.com/2016/01/create-customize-wordpress-child-theme/

Target. (n.d.). Retrieved February 10, 2020, from https://www.target.com/

Appendix

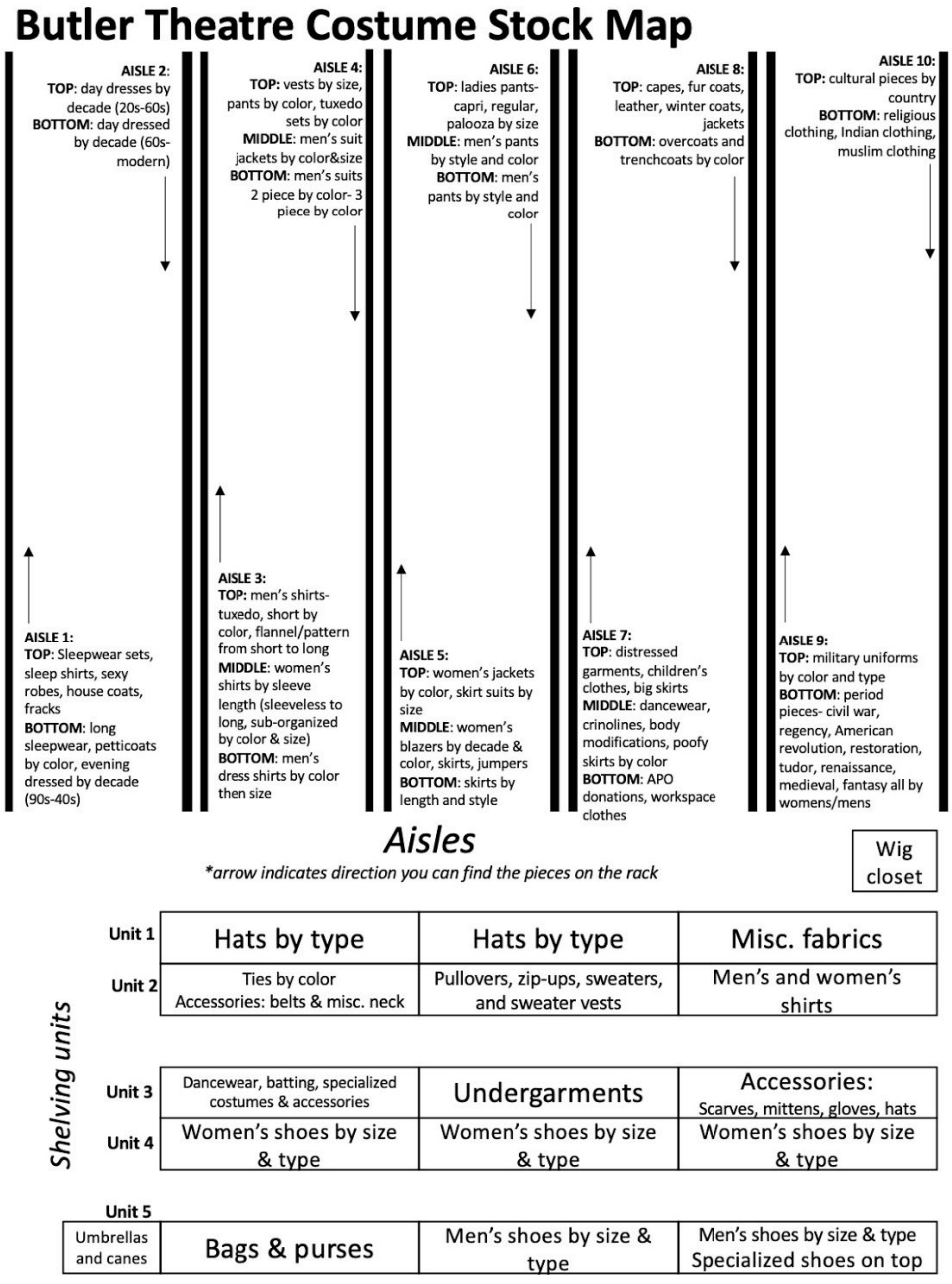

Figure A1: Costumes Stock Map

Shelving Unit Specifics

Unit 1
L to R
**based on facing the shelf*

Misc. Fabrics	Hats by type	Hats by type
Bali wraps, shoulder pieces	Academic hats, doctors masks, beanies	Tophats, bowlers, drovers, porkpie, derby
Brimless hats	Cultural/historical hats	Fedoras, newsboys, walking hats, bucket, baseball, trucker
Brimless hats	Occupational/event hats	Antique hats, specialty masks, earmuffs, fedoras
Brimless hats	Specialty hats	Military hats, hillbilly, leather

Unit 2
L to R

Ties/accessories	Pullovers, sweaters, etc.	Men's and Women's' shirts
Ascots, paper collars, caravats, jabots, dickies, scarves	Sweatpants, soccer uniforms, track suits, women's shorts	Women's sweaters, dusters, cardigans
Colored and pattern ties, special neckwear, bandanas, pocket squares	Men's pullovers, zip ups, velour tops, fleece jackets, elastic pants	Warm plain shirts, logo t-shirts, men's sleeveless, pullovers
Belts, sashes, cummerbunds	Men's cardigans, sweater vests, pullovers, polo shirts	Neutral/black plain shirts, turtlenecks, long sleeve, cardigans
Cultural masks	Men's shorts, women's shorts, women's pants	Henley shirts, long sleeve t-shirts, knit blouses, long underwear

Unit 3
L to R

Accessories	Undergarments	Dancewear, specialized, etc.
Winter mufflers & scarves, fingerless gloves, dress gloves, specialty gloves	Boxed pantyhose, men's athletic socks & tights, stretch pants, leggings	Ponchos, sewing patterns, parachute, batting
Silk and knitted scarves, mufflers, work gloves	Slips, briefs, boxers, bloomers, tights, tap pants	Feather boas, rompers, coveralls, undershirts, dancewear, chainmail
Men's opera scarves, women's shawls, scarves	Camisoles, tank tops, slips, lingerie, aprons	Veils, hospital gowns, sarongs
Women's gloves, shawls, scarves, wraps, mufflers	Corsets, bustiers, waist cinchers	Metal corsets, antique satin, fur pieces, kneepads

Unit 4
L to R

Women's shoes	Women's shoes	Women's shoes
Fencing mask, emotional masks, misc. masks	ankle boots (size 7), heels & loafers (size 11), drapery, blankets, bedding	Size 6 ½ or less: slippers, heels, galoshes, spats (any size), linens, armbands
Size 8-8 ½:slippers, flats, sandals, heels/ tap, ballet, jazz (any size)	Size 7-7 ½: Granny, sandals, slippers, tall boots, heels	Size 6 ½ or less: high heels, peep toe, flats, flip flops, boots
Size 8-8 ½: boots, clogs, platforms	Size 7- 7 ½: tennis shoes, flats, boots, square heels, sandals	Size 6 ½ or less: flats, loafers, peep toe, kitten heel, platforms
Size 8-10 ½: loafers, granny, boots, character shoes	Size 9-10 ½: square heels, flats, sandals, wedges, loafers, granny	Size 9-9 ½: boots, thin heels, wedges

Unit 5
L to R

Men's shoes	Men's shoes	Handbags & purses
Roller-skates, big prop boots, Japanese shoes	Size 10-12: rubber boots, dress shoes White and brown handbags	Mosquito netting, angel wings, bird puppets, specialty bags, straw bags
Size 7-10 ½: tennis shoes, tie shoes	Size 10-14: boots, tie shoes, slip ons	Purses, large handbags, clutches, army bags, canvas bags
Size 8-10 ½: sandals, tennis shoes, boots, slip-ons	Slippers, riding boots, felt boots (any size), shoes (13-13 ½)	Purses, colored handbags, maracas
Size 8-10 ½: boots, tennis shoes, tie shoes, slip-ons	Size 11-11 ½: tennis shoes, tie shoes, boots, slip ons, sandals	Carpet bags, doctors bags, messenger bags, luggage

**In the index, from top to bottom the selves are referred to by number. (Top shelf= 1st shelf, 2nd from top = 2nd shelf, etc.)

Figure A2: Costume Stock Map Shelving Units

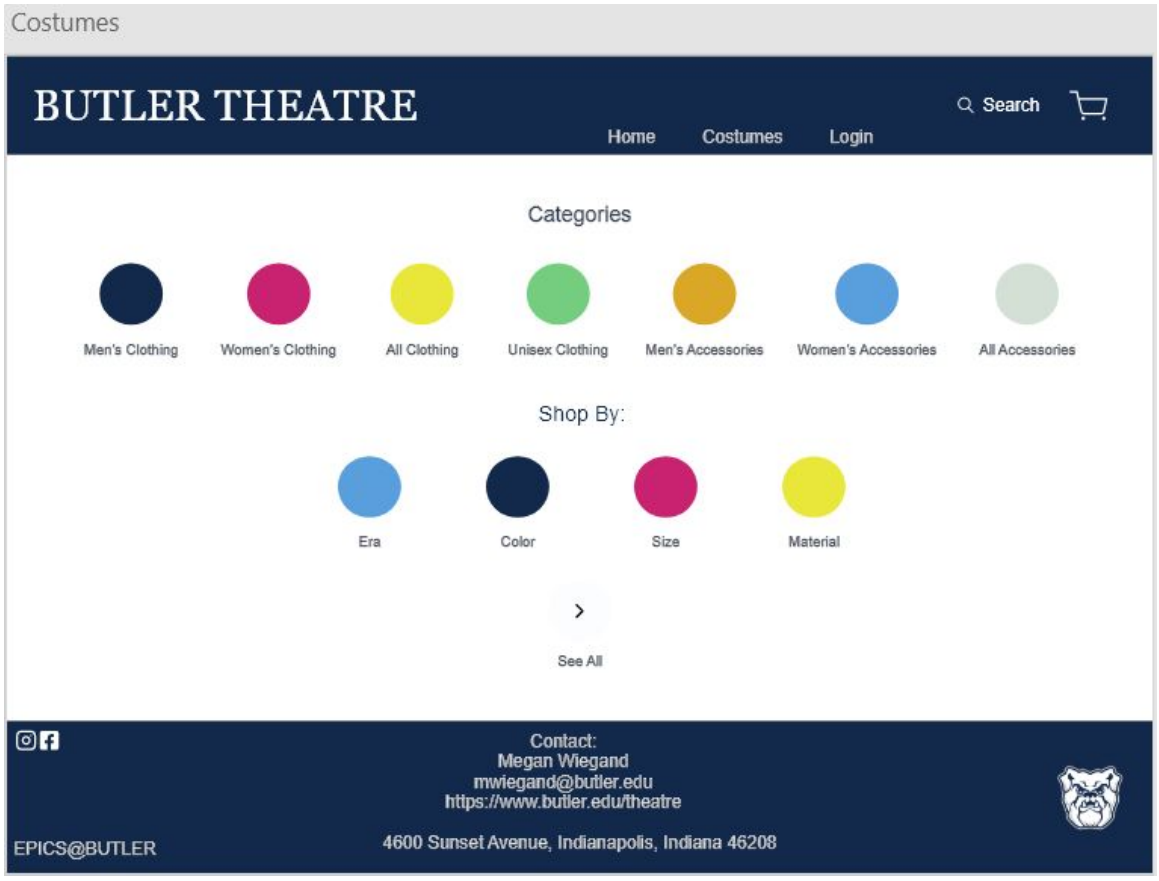

Figure A3: Costumes page of the wireframe design

Figure A4: Costume page of the wireframe design