Pocket Shelves

by

Zachary Abell, Adam Jackson, and Alexander Napper

Submitted to
the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

Zachary Abell	04/29/19
Zachary Abell	Date
Adam Jackson	04/29/19
Adam Jackson	Date
Alexander Napper	04/29/19
Alexander Napper	Date
Ryan Moore, Faculty Advisor	04/29/19
Ryan Moore, Faculty Advisor	Date

University of Cincinnati
College of
Education, Criminal Justice, and Human Services

April 2019

Pocket Shelves

Prepared by
Zachary Abell, Adam Jackson,
and Alexander Napper

Students of
University of Cincinnati
College of Education, Criminal Justice, and Human Services
School of Information Technology

April 29, 2019

Table of Contents

List of Illustrations

Tables

Figures

Acronyms and Abbreviations

API	Application Programming Interface
AWS	Amazon Web Service
IP	Internet Protocol
UI	User Interface
SSH	Secure Shell

Abstract

Our web application “Pocket Shelves” ensures small business owners save money by simplifying inventory management for our users. According to Small Business Trends, it’s estimated that businesses around the country lose over 300 billion dollars annually due to poor inventory management [1]. Pocket Shelves allows users to document and edit inventory live, as well as track and organize inventory physically with barcodes. Pocket Shelves also features analytics to help the user better manage their inventory financially. Pocket Shelves allows the user to share their information among an established group with easy to use permissions to restrict use. The user-interface of our application was built to work on any browser with the Bootstrap framework. The backbone of Pocket Shelves was written in Python 3, built with the Flask micro-framework, and primarily using PostgreSQL. Proper inventory management has never been easier, thanks to easy to use applications like Pocket Shelves.

Problem Statement

It is estimated that businesses lose 300 billion dollars annually, due to poor inventory management [1]. Having an accurate representation of one's inventory and sales can be a difficult and daunting task for those with new and small businesses. We think we have a solution that could be useful and helpful to many of these people.

Project Concept & Solution

Develop a mobile application that can be multifunctional for small businesses. The application will allow the user to add and modify inventory in real time. As well as have a barcode scanner and printer, for easy tracking and storing of physical items/goods. The application will also allow a group of users to make a 'community' where the information can be shared amongst those that are allowed to be involved. The community page will offer a messaging board as well as a board of inventory logs. The application will also be easily accessible, within almost any web browser as it will be formatted for desktops, tablets, and mobile devices.

This idea came from Adam, who stressed poor inventory management was a major problem in his prior jobs and COOPs. We all agreed after some research, with our own real life problem examples, and felt that we could create a solution together. We sought out to create an application that would not only be effective, but looked at as a valuable resource to any users who needs require this application and create a sense of community with colleagues.

Project Goals

- Allow user to add, delete, and edit inventory
- Allow user to print labels for physical inventory
- Allow user to create a community / group to share information with
- Allow user to see live analytics of their inventory
- Allow user to access web application easily off of any device

Overview

The remainder of this final report outlines in detail how the project was completed. The report includes the following sections: design objectives, methodology, budget, timeline, technical elements, and concluding with learned lessons along with future plans.

Design Objectives

The primary objective of the application is to create an interactive web application to enhance the user's inventory management. The user will be able to add, delete, and edit inventory. Additionally, the user will be able to print barcodes in order to manage physical inventory. The user will also have access to data analytics per item inventorized.

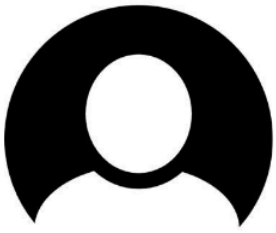
The application's UI is meant to be clean, simplistic, and modern built off of Bootstrap framework. Using this technology will help to deliver a fluid UI that is appealing to the eye and not a chore to navigate through. The application is meant to provide the user with all of the information they would expect to find, in each page, and offer the ability to navigate freely and easily by using the nav bar.

The backend will be supplemental to the frontend, as it will ensure that the user's actions in the application are accounted for accurately. The backend will be written in Python 3 to ensure that the user's information is parsed and saved correctly. JavaScript and AJAX will help to link our database to the front end of the application.

User Profile

Figure 1: User Profile, illustrates the user profile for the Pocket Shelves application. It denotes potential users of the application, related experience and similar applications that the user may have encountered, the tasks the user is expected complete when operating Pocket Shelves, the expected use of Pocket Shelves per person, and key interface design requirements Pocket shelves will need to optimize the user experience.

User Profile



Application

Pocket Shelves

Potential Users

- Business Owners
- Merchandise Collectors
- General Consumers

Software, Interface, and Related Experience

All Pocket Shelves users should have a basic understanding with a computer and mobile device, as well as navigating through a website UI. A basic understanding of inventory management would be effective as well.

Experience with Similar Applications

- Fishbowl Inventory
- Lead Commerce
- Agiliron Inventory Management System
- Systum
- Trxio

Task Experience

At the initial opening of the web application, the user will be prompted with a login screen. After logging in, the user will see their inventory, community their apart of, and settings to personalize their profile within the application. Clicking on the inventory will provide a wide variety of functions for the user to engage in with their inventory. The community page will provide them with information shared from other users within their community.

Frequency of Use

This application is intended to be used when the user needs to add/remove/edit inventory. This would be the most common use, but other features such as printing barcodes for physical inventory management and engaging on the community page would be a commonly used as well.

Key Interface Design Requirements that the Profile Suggests

- Fluid/intuitive UI
- Fluid UI compliant with multiple desktop and mobile devices
- Appealing design
- Easy to read and follow headers and text formatting
- Attractive layout

Figure 1. User Profile

Use Case Diagram

The following diagram, Figure 2: Use Case Diagram, displays the use case for Thrive. The Diagram depicts all users of Thrive along with the corresponding tasks each user will have access to when interacting with the application.

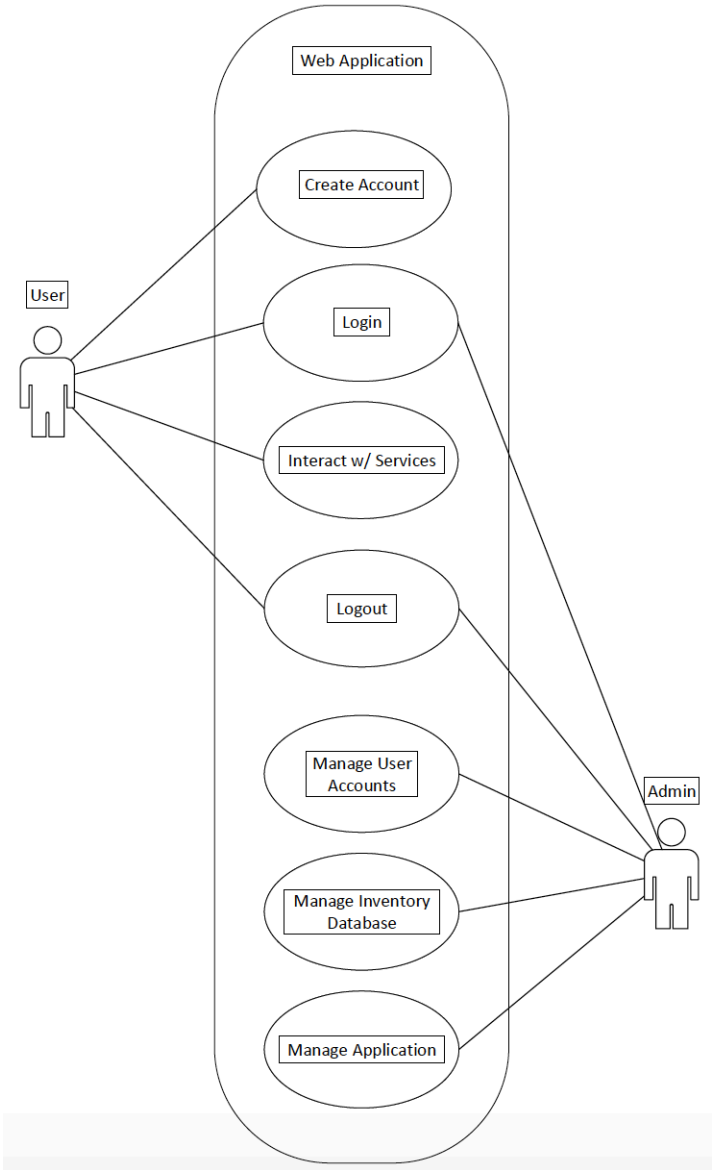


Figure 2 : Use Case Diagram

Budget

Table 1: Project Budget outlines the budget for this project. The expenses are organized into two categories: materials and labor. The backend will be hosted using Amazon Web Service's (AWS) with a free trial. All frameworks and development tools used to create the application are free and open source. The predicted wage cost totaled $27,000 under the assumption that each team member would earn $25 per hour. However, the actual cost of this project is $0 since this project is being completed for the procurement of a Bachelor's of Science degree in Information Technology.

CATEGORY	ITEM	DESCRIPTION	EXPECTED COST	ACTUAL COST
Materials	**Hardware**	**Computer servers and devices used to create, test, run, and deploy the application.**	**$0**	**TBD**
	Software	**Frameworks and development tools used to develop the application.**	**$0**	**TBD**
Labor	**Conferences and Meetings**	**Funding for employee collaboration, learning events, and meetings.**	**$0**	**TBD**
	Actual Wage Costs	**The actual wages that will be distributed for development of the project.**	**$0**	**TBD**

	Simulated Wage Costs*	**The predicted wages. (We will assume 3 developers making $25 USD / hour).**	**$27,000**	**TBD**
TOTALS			**$0**	**TBD**
Simulated Wage Costs are not included in the cost totals.				

Table 1: Project Budget

Project Schedule

Figure 3-8: Gantt Charts and Table 2: Work Breakdown Structure outline the projected schedule for completion of this project.

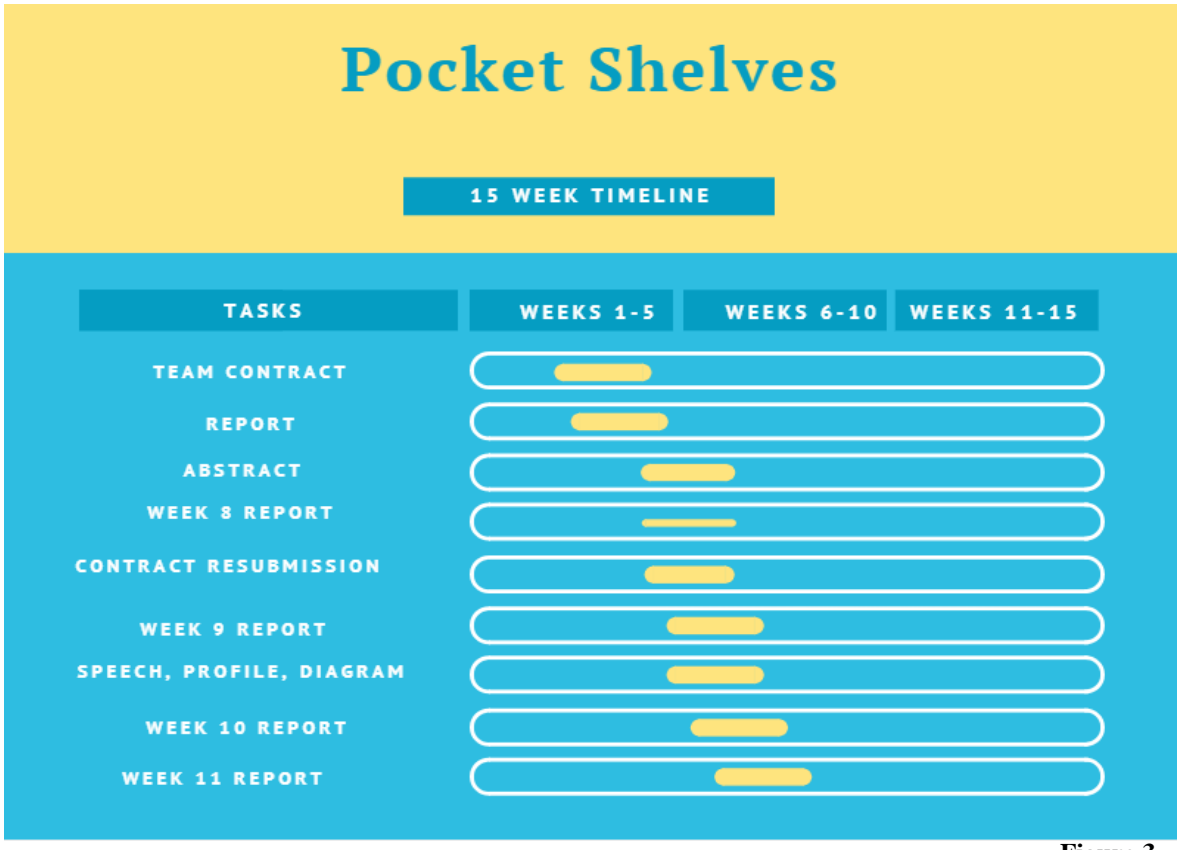


Figure 3

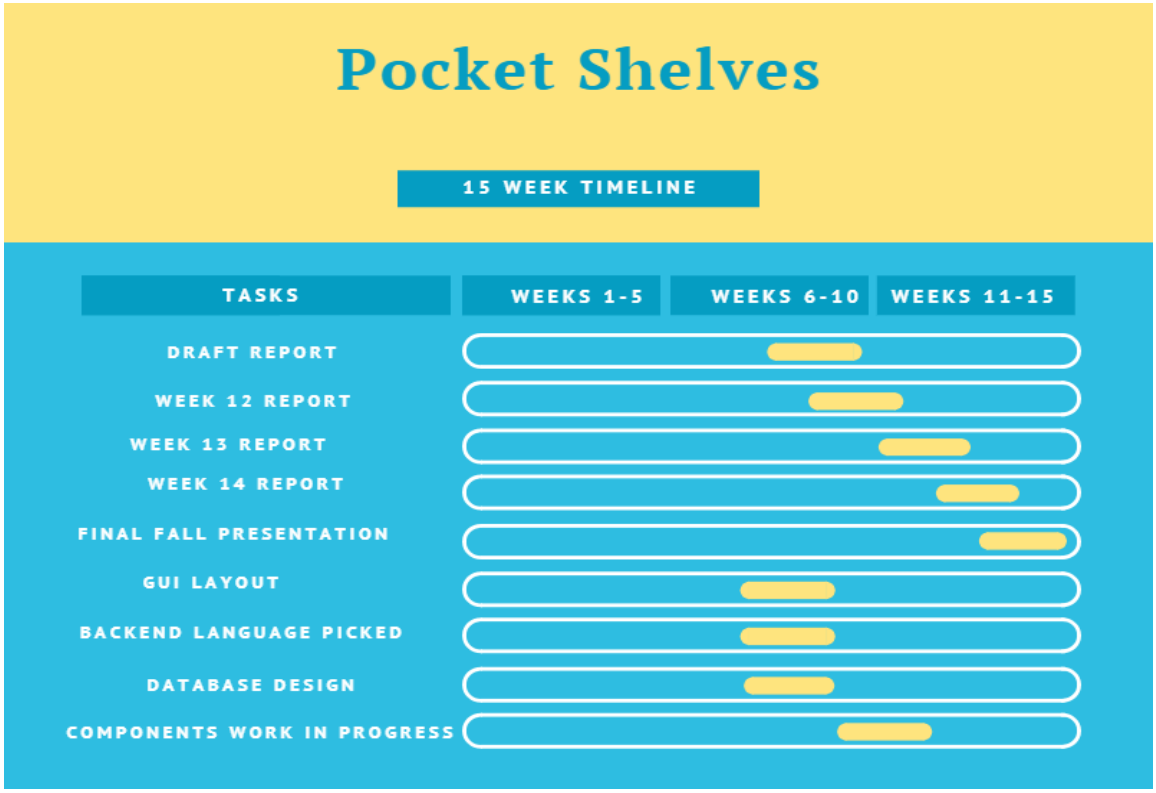


Figure 4

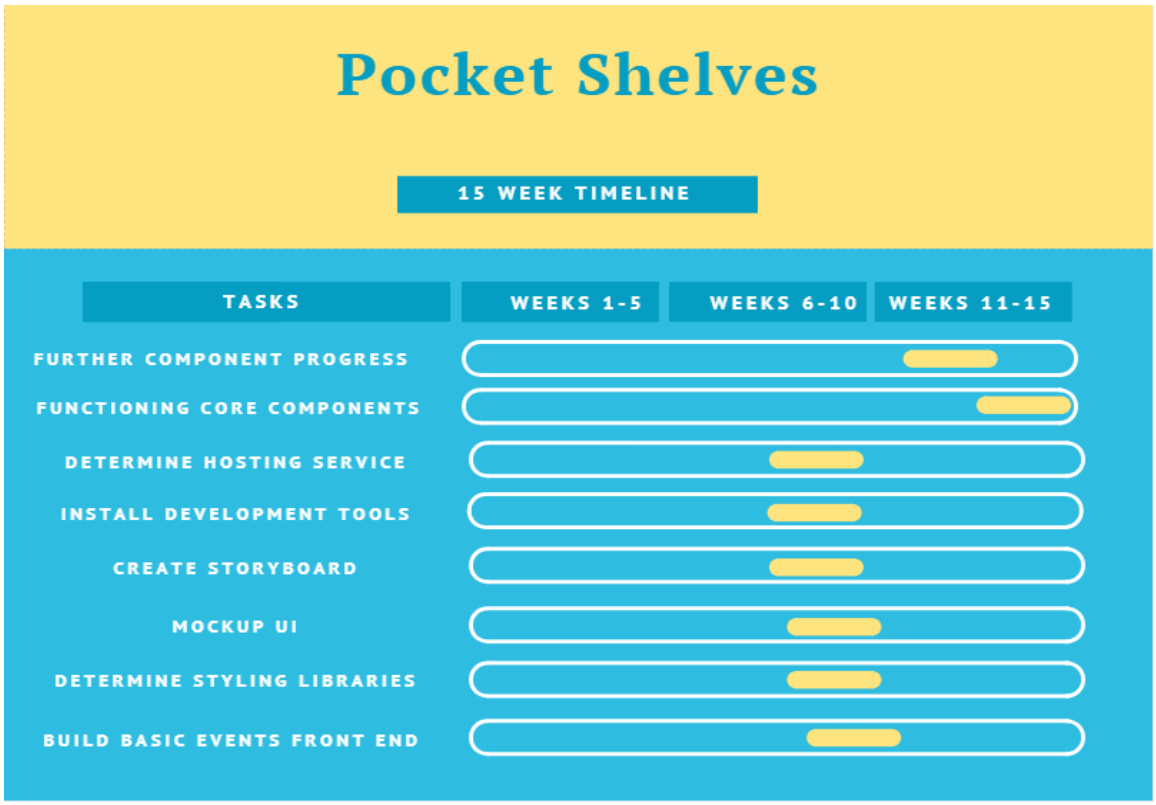


Figure 5

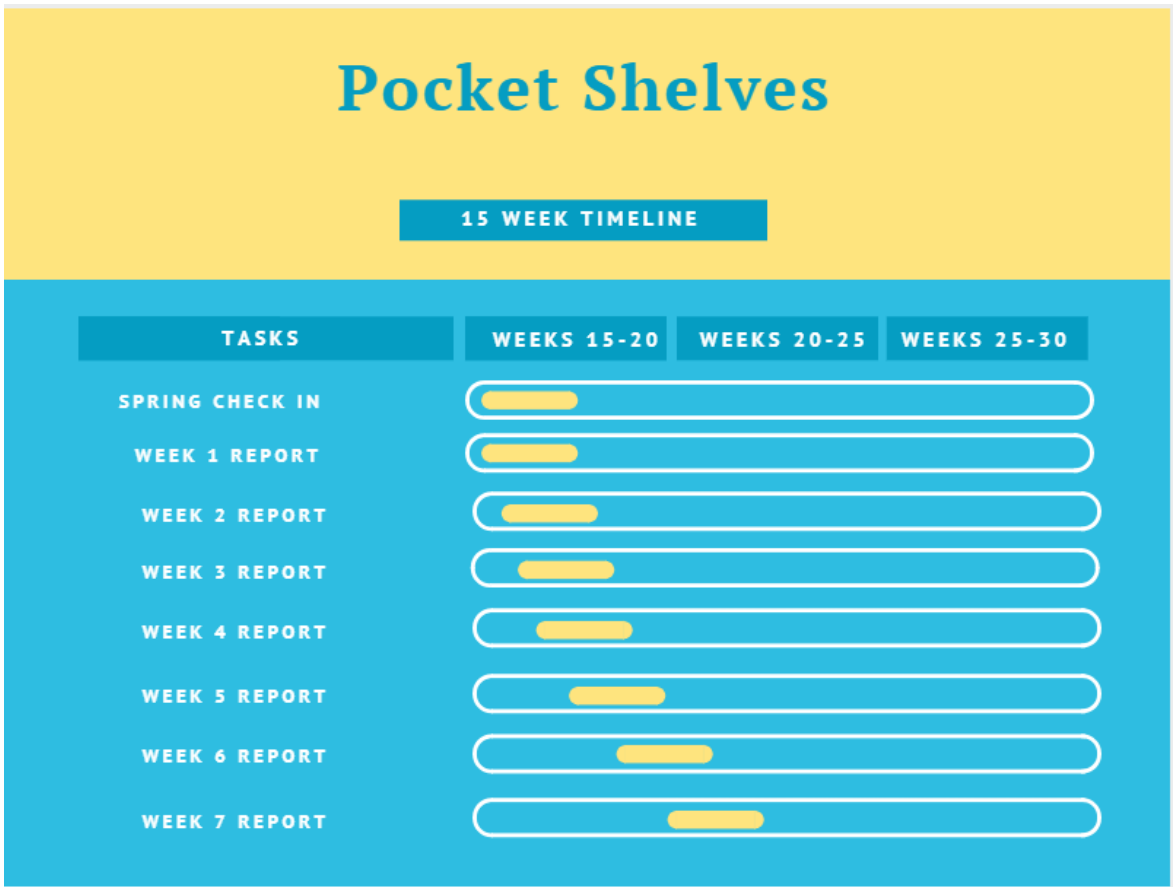


Figure 6

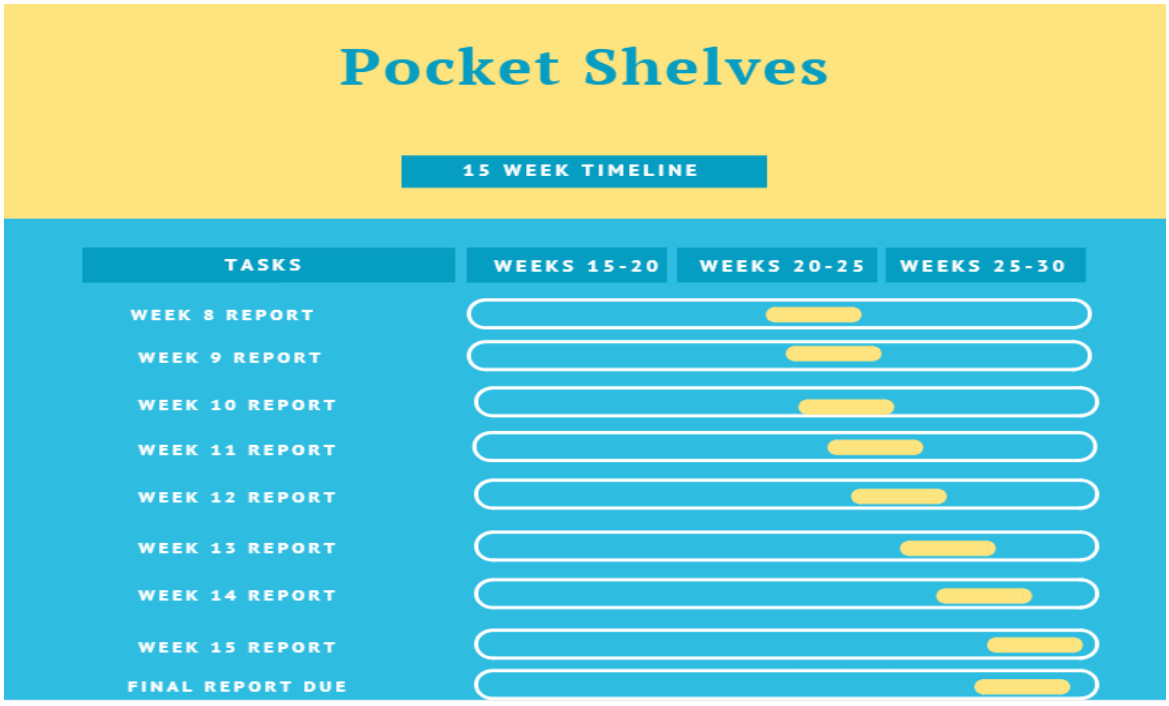


Figure 7

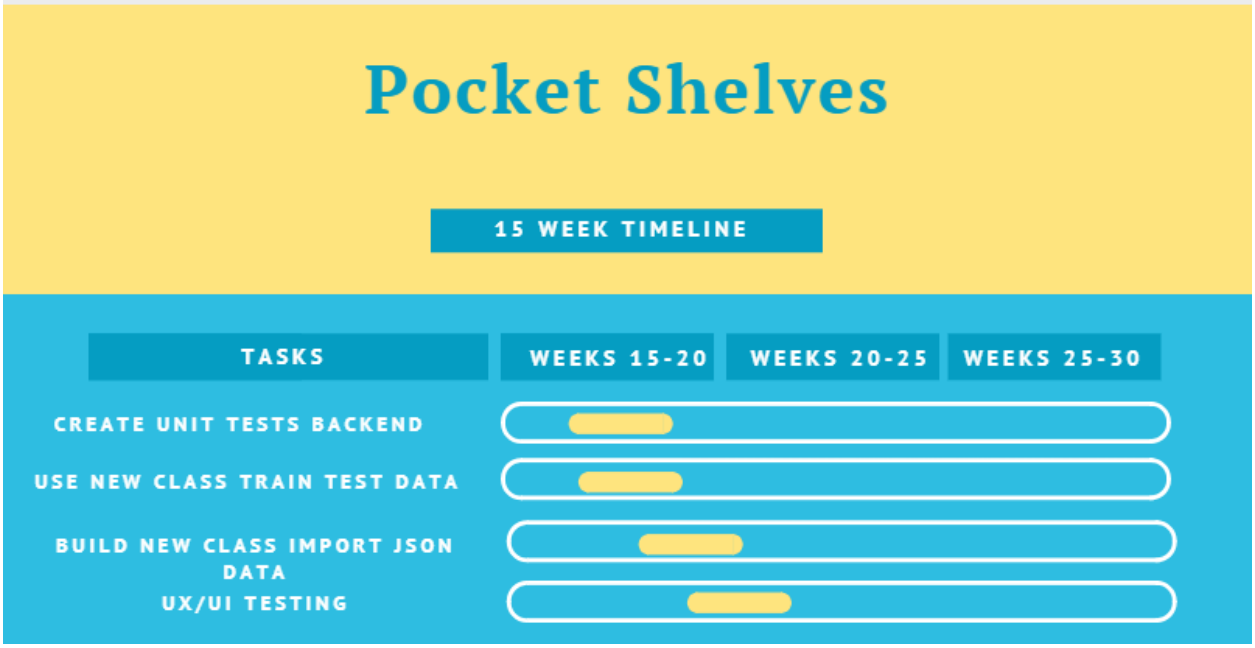


Figure 8 : Gantt Chart

WBS

The following table, Table 2: Pocket Shelves WBS is a table that represents the work schedule that shows deadlines for stretch goals in the project.

Pocket Shelves WBS			
TART DATE:	END DATE:	DESCRIPTION:	DURATION (days):
9/3/2018	**4/15/2019**	**1. Project Management**	**223**
9/3/2018	9/24/2018	1.1 Team Contract DUE	21
9/23/2018	9/29/2018	1.2 Report DUE	6
10/8/2018	10/15/2018	1.3 Project Abstract DUE	7
10/8/2018	10/15/2018	1.4 Week 8 Report DUE	7
10/8/2018	10/15/2018	1.5 Resubmission of Team Contract	7
10/16/2018	10/22/2018	1.6 Week 9 Report DUE	6
10/15/2018	10/22/2018	.7 Elevator Speech, User Profile Report/Table, Use Case Diagram	7
10/23/2018	10/29/2018	1.8 Week 10 Report DUE	6

10/30/2018	11/5/2018	1.10 Week 11 Report DUE	6
10/29/2018	11/5/2018	1.11 Draft Report DUE	7
11/6/2018	11/12/2018	1.12 Week 12 Report DUE	6
11/13/2018	11/19/2018	1.13 Week 13 Report DUE	6
11/19/2018	11/26/2018	1.14 Final Fall Presentation	0
11/20/2018	11/26/2018	1.15 Week 14 Report DUE	6
1/7/2018	1/13/2018	1.16 Spring Check-In	6
1/14/2019	1/20/2019	1.17 SS Week 1 Report DUE	6
1/21/2019	1/28/2019	1.18 SS Week 2 Report DUE	6
1/29/2019	2/4/2019	1.19 SS Week 3 Report DUE	6
2/5/2019	2/11/2019	1.20 SS Week 4 Report DUE	6
2/12/2019	2/18/2019	1.21 SS Week 5 Report DUE	6
2/19/2019	2/25/2019	1.22 SS Week 6 Report DUE	6
2/26/2019	3/4/2019	1.23 SS Week 7 Report DUE	6
3/5/2019	3/11/2019	1.24 SS Week 8 Report DUE	6
3/12/2019	3/18/2019	1.25 SS Week 9 Report DUE	6
3/19/2019	3/25/2019	1.26 SS Week 10 Report DUE	6
3/26/2019	4/1/2019	1.27 SS Week 11 Report DUE	6
4/2/2019	4/8/2019	1.28 SS Week 12 Report DUE	6
3/15/2018	4/15/2018	1.29 Final Report DUE	28
4/9/2019	4/15/2019	1.30 SS Week 13 Report DUE	6
4/9/2018	4/9/2018	1.31 IT Expo	0
10/1/2018	**10/13/2018**	**2. Research**	**13**
10/1/2018	10/13/2018	2.1 GUI Layout Design	13
10/1/2018	10/13/2018	2.2 Backend Language Picked	13
10/1/2018	10/13/2018	2.3 Database Diagram	13
10/14/2018	**11/24/2018**	**3. Front End Development**	**41**
10/14/2018	10/20/2018	3.1 Work in progress GUI	6
10/21/2018	10/27/2018	3.2 Further Progress Made	6
10/28/2018	11/24/18	3.3 Functioning Core Components	27

10/14/2018	**1/28/2019**	**4. Back End Development**	**105**
10/14/2018	10/20/2018	4.1 Partial Backend Functionality Design	6
10/14/2018	10/20/2018	4.2 Partially Functioning Database Design	6
10/21/2018	10/27/2018	4.3 Further Progress Made	6
10/28/2018	11/24/18	4.3 Functioning Core Components	27
11/25/18	1/28/18	4.4 Polishing Core Components	21
10/16/18	**10/30/18**	**5. Environment Setup**	**13**
10/16/18	10/17/18	5.1 Determine hosting service	1
10/17/18	10/22/18	5.2 Install development tools for all team members	5
10/23/18	10/30/18	5.3 Build Infrastructure	7
10/16/18	**11/11/2018**	**6. Front End Development**	**24**
10/16/18	10/21/18	6.1 Create Storyboard	5
10/21/2018	10/27/2018	6.2 Mockup UI	6
10/27/2018	10/28/2018	6.3 Determine Styling Libraries	1
10/29/2018	11/2/2018	6.4 Build basic events for front end	5
11/3/2018	11/7/2018	6.5 Create landing page	4
11/8/2018	11/11/2018	6.6 Create login/sign up page	3
1/29/2019	**3/19/2019**	**7. Testing**	**50**
1/29/2019	2/5/2019	7.1 Create unit tests for backend	7
2/6/2019	2/13/2019	7.2 Use new class to train the test data	7
2/14/2019	2/19/2019	7.3 Build new class to import Json data	5
1/29/2019	3/19/2018	7.4 UX/UI testing	50

Table 2: Work Breakdown

Technical Elements

An application's success will ultimately be decided on how well the design is. It is critical to have a strong design, front-end and back-end, as missing the mark will result in failure.

Network

Pocket Shelves will be hosted on AWS. It was an easy choice, as it was free and scalable with the trial offering more than enough amenities. We have one root account, and then additional individual accounts to access the server.

Application

The front-end will have been developed and written in HTML5 as well as JavaScript. The front-end application will be using the Bootstrap framework for styling. Bootstrap allows the application to scale to all devices (desktop, tablet, mobile) while also offering a wide variety of tools to create a visually appealing application.

The backend of pocket shelves is written in Python 3 and makes use of the Flask microframework for ease of development and simplicity. We make use of multiple Flask plugins/libraries such as Flask-login, Flask-sqlalchemy, and Flask-wtf.

Database

Pocket Shelves will make use of PostgreSQL for its database. By making use of sqlalchemy as the database ORM, a customer hosted instance of Pocket Shelves will be able to make use of a wide variety of other databases such as Oracle, MySQL, and SQLite. Pocket Shelves uses AJAX and jQuery to store/POST data from the user on the front-end, as well as to GET data to produce for the front end analytics

User Interface

Below, are brief descriptions and images of Pocket Shelves current working pages and UI elements. All of these elements were created with Bootstrap.

Login Page

The following image, Figure 9: The login screen is meant to be very simple and will be the first thing the user sees upon requesting access to the site. The user will always be prompted to login back in, upon revisiting the site. The picture below represents the login screen.

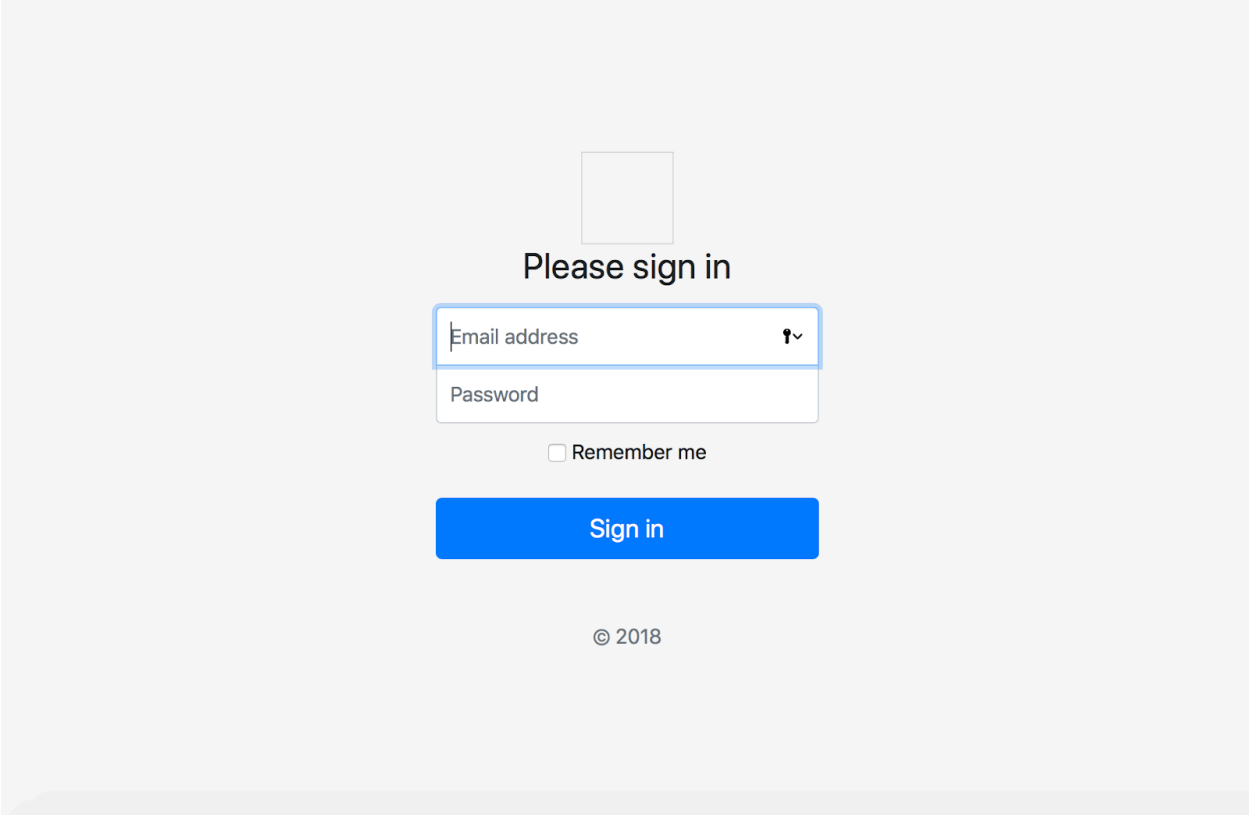


Figure 9: Login Screen

Inventory Page

The following image, Figure 10: The inventory page will be a live representation of all the inventory that the user has entered into the application. The user will see all entries, with critical information being in plain site. Additionally, the 'edit' button will be located here if the user wishes to edit an inventoried item.

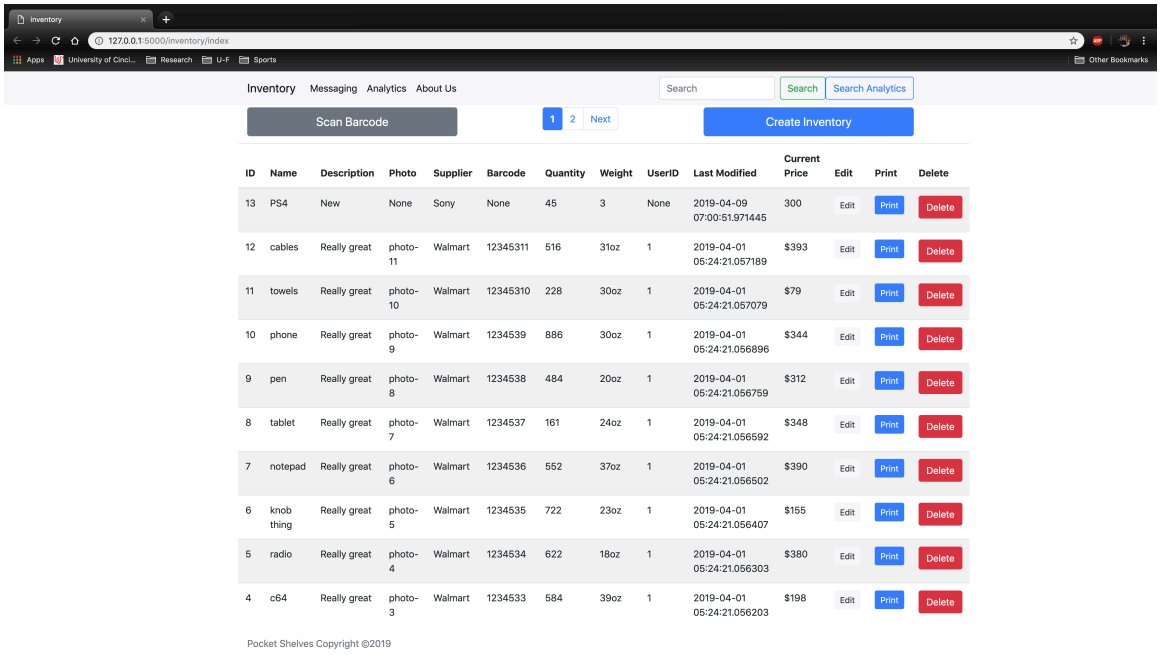


Figure 10: Inventory Screen (Desktop)

Nav Bar

The following image, Figure 11: The nav bar will be accessible on each page within the site. The goal was to make it so the nav bar could offer everything the user would need, with a limited amount of clicks. The user will have the ability to add inventory, access their 'community', and check analytics.

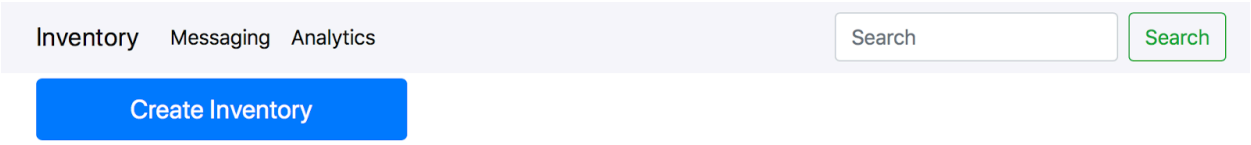


Figure 11: Nav Bar

Create Inventory

The following image, Figure 12: The create inventory form is accessible on the inventory page of Pocket Shelves. It offer the ability to create inventory by assigning a name, description, picture, supplier, quantity, weight, and price. Upon filling the form out, hitting the submit button will produce a new row in the inventory selection with the entry the user has entered.

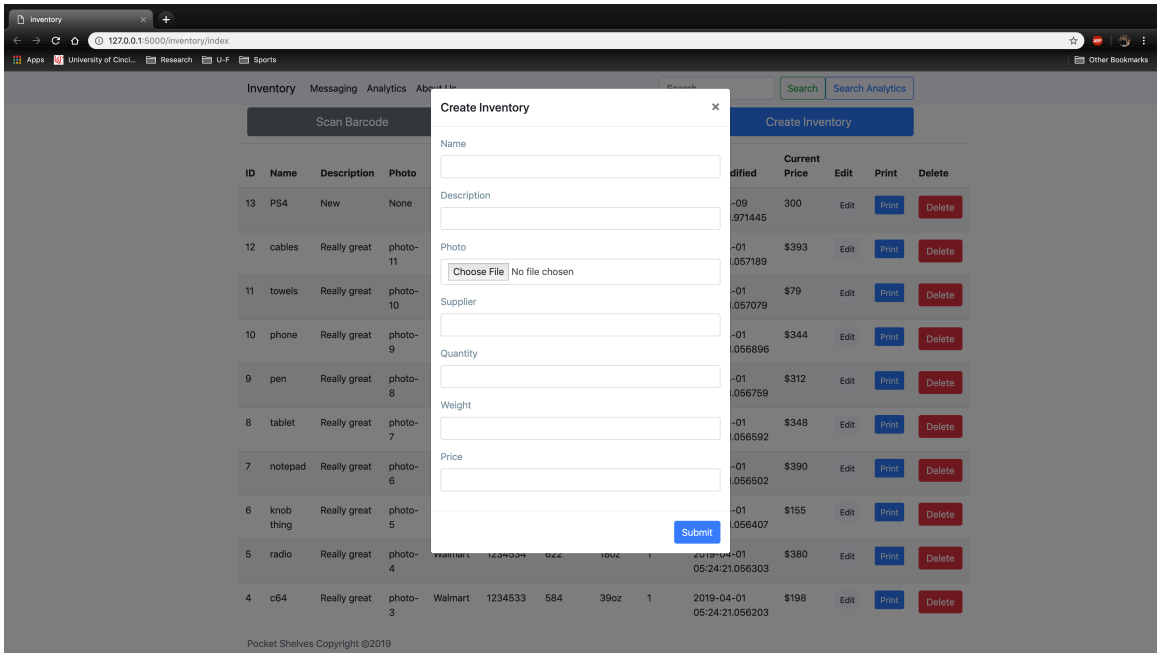


Figure 12: Create Inventory Form

Analytics

The following image, Figure 13: The analytics page is accessible through the nav bar found on every page. Upon opening the page, the user will find all of the current inventory they have logged. It has basic info in the column, with an option to click the analysis button for more detailed logging information with things like date added, date modified, as well other things you'd find on the inventory page. There is also a global analytics button will display a series of graphs with more general analytics such as all items and quantity and what is moving on a day to day or week to week basis.

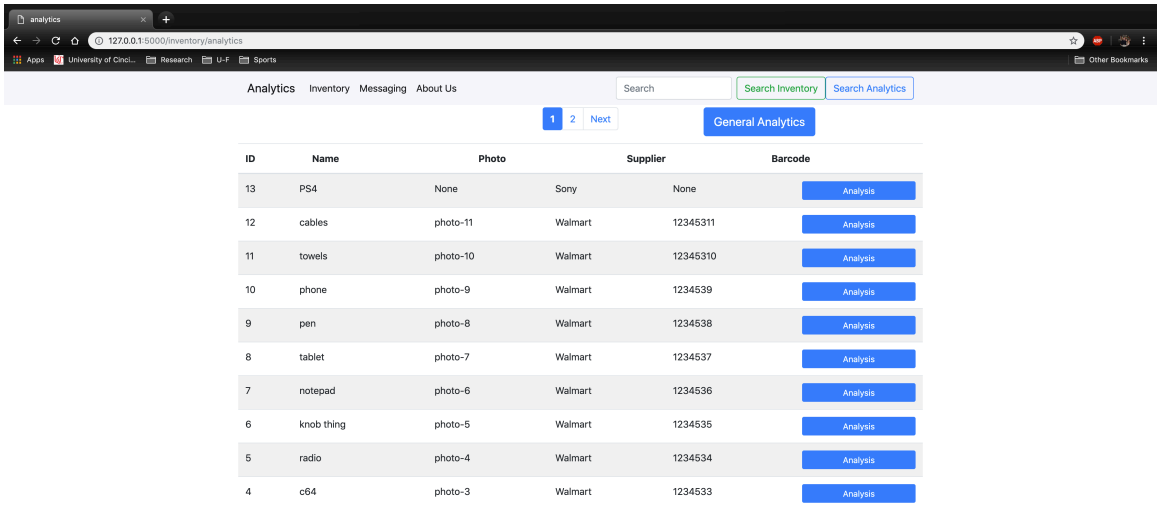


Figure 13: Analytics Page

Community Page

The following image, Figure 14: The community page is accessible through the nav bar found on every page. The community page will allow a group of user to share thoughts on a message board. Additionally, the users will be able to see logging information on inventory, with things like *insert name* modified item: Xbox.

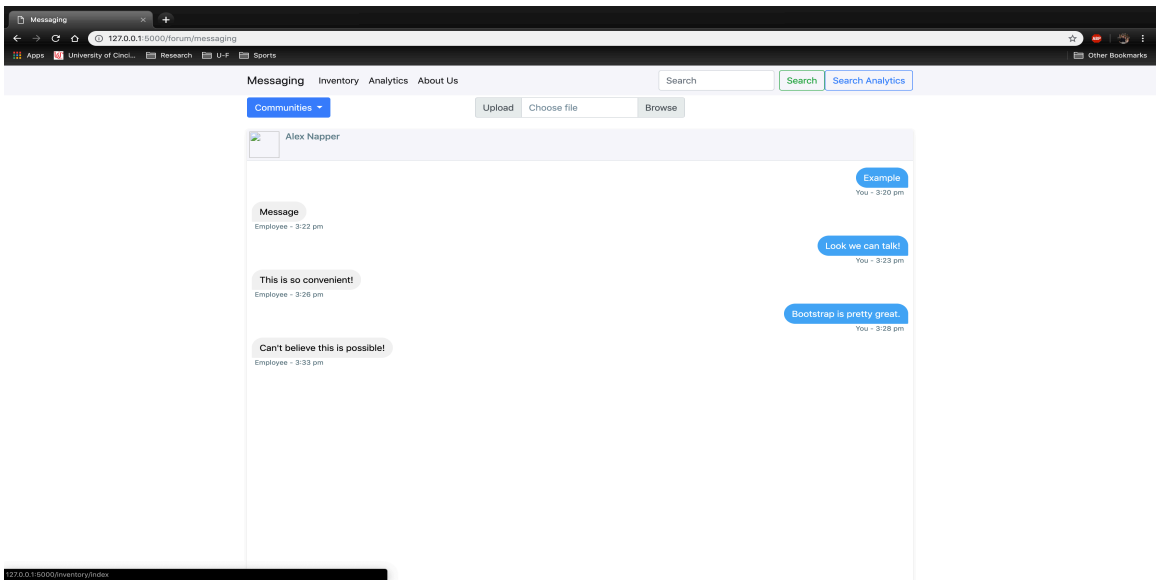


Figure 14: Community Page

About Us Page

The following image, Figure 15: The About Us page is accessible through the nav bar found on every page. The about us page will is just general information about the team built Pocket Shelves.

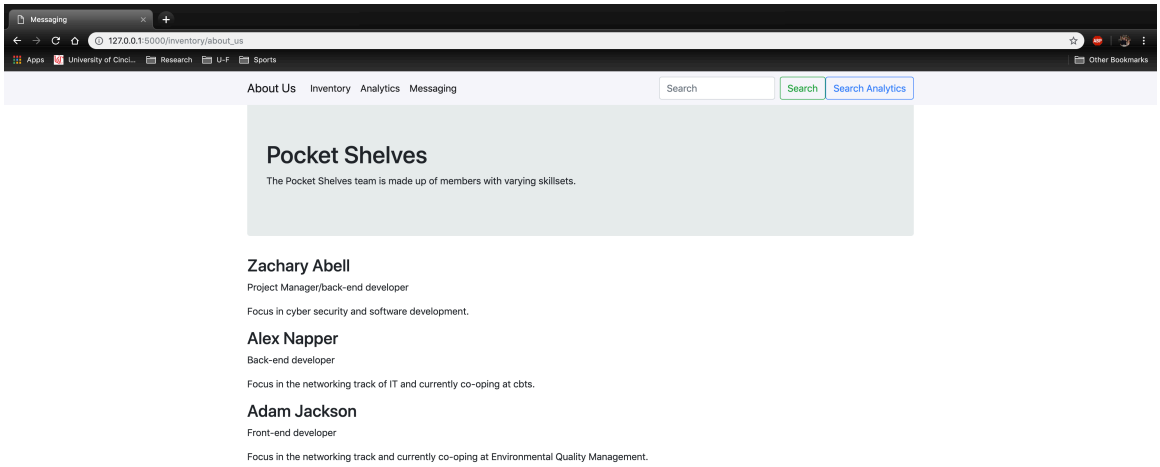


Figure 15: About Us Page

Testing

This section of the paper will delineate the methodologies and testing tools for the Pocket Shelves web application (can be run on iOS, Android, and desktop browsers). This should be used as a guide as this section of the paper will provide a testing breakdown as well as the features that have been tested and how they were individually tested.

Scope

The scope of this testing is to test the Pocket Shelves' website and database to ensure each endpoint is returning proper results and making proper changes to the backend database. There is two sections for our testing: The user site used by those visiting the Pocket Shelves' website and the API that exposes similar functionality to automated tools. This testing will let us know if any issues were introduced during development or if there is something already out of spec that has not been noticed.

Objectives

The purpose of our testing is to ensure the inventory management system we built operates with the highest levels of usability, performance, and reliability. The testing process will allow us to identify any bugs or malfunctions in the web app before going live with the site. In doing so, the client will have a more stable and user friendly environment to work in.

Logging Tests and Procedures

Because of our use of the Flask Microframework, testing each page was straightforward.

1. UI Test — For every page we are able to insert data into a test SQLite database then run the function used to generate the page and compare its output against a known good page. These two pages can then be compared to see if there was any unintentional or intentional changes.
2. API Test — Each API endpoint can be queried similar to the UI test and the JSON returned is compared against known good results to see if there was any unintentional or intentional changes.
3. Database Test — Posting to API endpoints to ensure proper changes are made to the backend database. Eg. items are added.

Pass/Fail Conditions

Pocket Shelves is expected to pass all tests during development, any failed tests mean something on the pages is not being generated correctly or the database is not being properly modified. Any failures in testing should be raised as an issue to be looked into.

Lessons Learned

The lessons learned during the testing of the Pocket Shelves web application were not as plentiful as anticipated, but also still very important. There were instances that were quite easy to test such as testing UI components (scaling the page size/window, testing links, testing buttons, and ensuring content is displayed properly). The instances where it was more difficult to test for were more on the backend of the web application. We had to test to ensure that all the data entries were being properly stored in a database, able to be viewed, and able to be edited properly. This makes for a very strict set of conditional statements that must be created to ensure accuracy during these testing processes. The biggest lesson learned is that anything and everything must be tested. This is not something we did not already know, but going through your own work, and testing every bit of it, will be a humbling and educational experience.

Conclusion

The lessons learned, through the end of the project, mainly revolve around us learning more about front end and back end systems as well as working together. We have all learned a great deal about the technical sides of the project. Adam, had a limited software background, has learned a great deal about front end systems and working with Bootstrap. Alex, has been learning to apply his software background through our project, while also being a great resource to the rest of the team. Zachary has learned about managing a project and group of people and just how difficult it can be as well as more about databasing with front end systems. All of us, are learning how to work as a unit and how to accommodate each other's schedules as well as work through problems together.

References

[1] DashBurst. (2017, November 02). Proper Inventory Management Means Customers Get What They Want and When. Retrieved from https://smallbiztrends.com/2016/06/inventory-management-issues.html