Book It
by
Jeremy Grice, Anderson Stoffer, Deonte Williams

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

Jeremy Grice	4/30/2019
Jeremy Grice	Date
Anderson Stoffer	4/30/2019
Anderson Stoffer	Date
Deonte Williams	4/30/2019
Deonte Williams	Date
Abdou Fall	4/30/2019
Abdou Fall, Faculty Advisor	Date

University of Cincinnati
College of
Education, Criminal Justice, and Human Services
April 2019

Book It

Prepared by
Jeremy Grice, Anderson Stoffer, Deonte Williams

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

April 15, 2019

TABLE OF CONTENTS

Illustrations

TABLES

FIGURES

ACRONYMS AND ABBREVIATIONS

API – Application Programming Interface
AWS – Amazon Web Service
JSON – JavaScript Object Notation

ABSTRACT

Students need reliable places to study. When students are successful with their studies, they do better in class and achieve higher grades. Currently, there is no single solution at the University of Cincinnati to locate and reserve places to study. This is why our team built Book It. We want to put the power back in the hands of students, by providing them with a single solution which gives them access to real time data. The Book It web application solves these problems by delivering a tool that allows students to connect and find quality spaces to study together.

PROBLEM STATEMENT

Introduction

For students, being able to locate a place of study can be a daunting task. Even more, locating a space on a student's respective campus and finding out if the location is available requires navigating numerous apps, and can sometimes leave a student empty handed at the end of the process. An application that can search for, then compare the results in real time is a need that is not currently available.

Problem

At the University of Cincinnati, there is currently not a single solution for finding a location to study on campus. This proves to be problematic for students. A 2015 study conducted by the Bureau of Labor Statistics show that students are spending roughly 3.3 hours per week engaged in educational activities. This is in sharp contrast to the long-held rule of thumb that students should be spending at least 2 hours per hour spent in class. While there are several factors in why students spend so little time studying, finding a location to study is one of the major ones. As studies have shown, poor grades can lead to a negative impact on a student's overall health. To combat this issue, certain steps can be taken to remove such stressors from student's life.

One method would be to use a student's universities website to search for libraries, as well as a location-based search tool such as Google maps, and finally utilizing another application to place the event in a calendar, and lastly sending out notifications to all the other students involved in the study session. This is overly cumbersome and not very practical, as it takes way too much effort and time. What these other numerous solutions do not offer is a way to

do it all in one single application, nor is there an application out there for searching for the study spaces located around a student's campus.

Solution

A solution that would resolve the problems stated in the section above would be to holistically assist the end user in finding a study space for their next study session. This assistance would take into consideration several factors, such as time, date, relativeness to outside amenities (restaurants, parking, etc.), and relativity to the campus. When taking these factors into consideration, the end user would be given a better idea of how to plan their study schedule for the foreseeable future. This would also allow students who may not have access to a personal vehicle the ability to plan their trip well in advance. This would save students a considerable amount of time, since they currently must couple several applications together to plan study sessions with other students.

Project Description

Book It will be a responsive web application that will allow students to search for and reserve study sessions with their fellow classmates. The application will query Databases and Application Programming Interfaces (API) and return the relevant information the users requested in a clean and easily useable tool. The ability to search for and find relevant data will put the power to succeed back in the hands of the students.

User Profile

Figure 1: User Profile, as shown below, illustrates the potential user profile. It brings together the related experience and similar applications that the user may have used in the past, the tasks the user is expected to complete when using the web application, the expected use and key interface design requirements.

User Profile

Book It

Users
Current college students

Software, Interface, and Related Experience
Book It is aimed at users who have experience with desktop devices, tablets, and/or mobile phones who have experience with mobile browsers and navigating through a guided calendar-centric experience.

Experience with similar applications
Book It users may have had experience with, but not limited to, the following applications:

- Airbnb
- Google Calendar
- iCalendar
- Facebook Events
- Fandango
- Eventbrite

Task Experience
The end user will follow a simplistic, guided systematic approach to finding a study space. They will be guided through a series of dropdown menus that dynamically generate with every choice in the dropdown prior. The user will have the ability to select a location based on their search criteria, and once they arrive at their final selection, they will be able to schedule an event and send invitations to the individuals in their group.

Frequency of Use:
The frequency of use will vary from individual-to-individual. Users will only use our application when they need to find a space to study.

Key Project Design Requirements that the Profile Suggests:

- Mobile responsive design that scales to any device
- Simple, user friendly User Interface
- Quick start up

Figure 1. User Profile

Use Case Diagram

Figure 2: Use Case Diagram, displays Book It and its use case. The Diagram depicts the users, along with the available tasks that the user will have access to when interacting with the application.

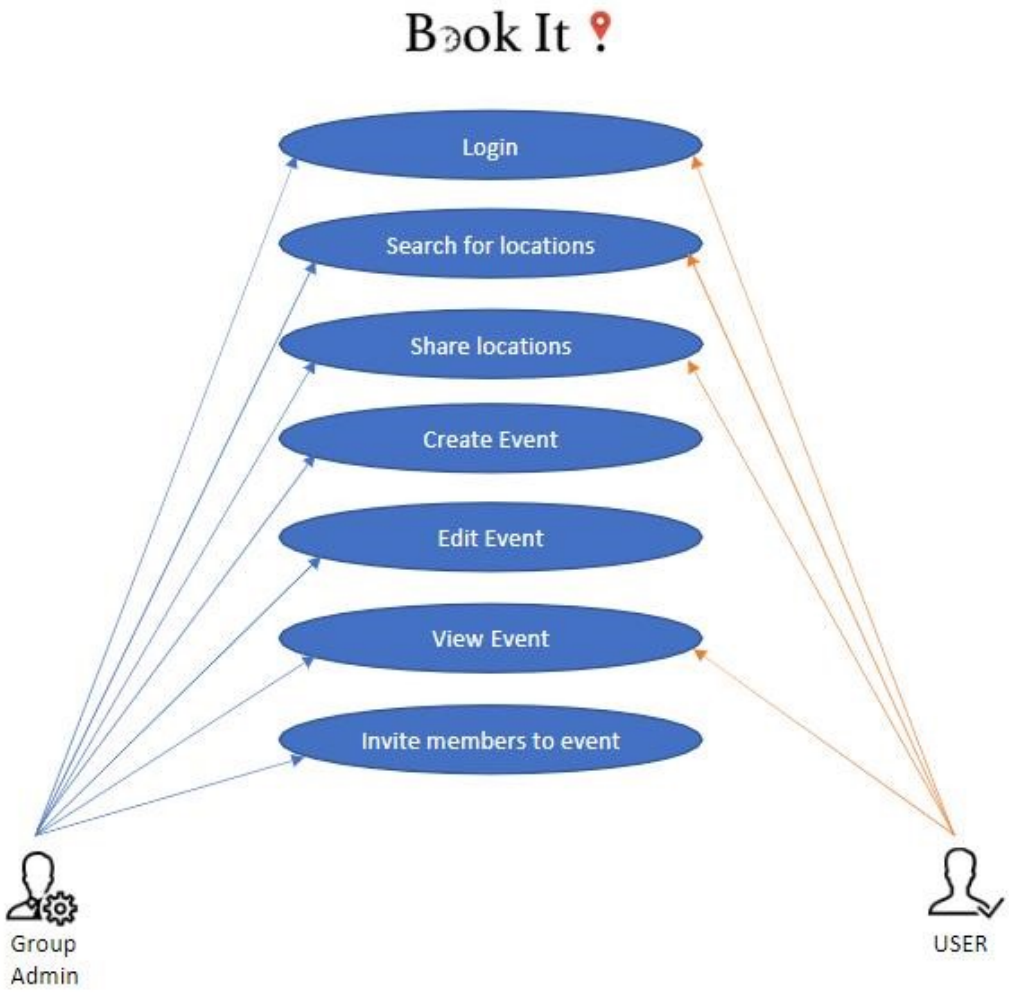

Figure 2. Use Case Diagram

PROJECT MANAGEMENT

Budget

Table 1: Project Budget outlines the budget for this project. The expenses are sorted into two categories: materials and labor. To keep our overall costs down, the backend will be hosted using Amazon Web Service's (AWS) free trial. All frameworks and development tools used to create the application are free and open source. The total actual cost of this project is $0 since this project is being completed as a requirement for the completion of a Bachelor of Science degree in Information Technology at the University of Cincinnati. Our budget did not change from one semester to the next. We kept everything free.

CATEGORY	*ITEM*	*DESCRIPTION*	*Expected/ 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	*Team Meetings*	*Funding for employee collaboration, and meetings.*	*$0 / TBD*

	Salary Cost	*This is the wages for the development of the web application.*	*$0 / TBD*
TOTALS			*$0 / TBD*

Table 1. Project Budget

Objectives & Deliverables

The main objective of this project is creating a deliverable product in the form of a fully responsive web application. This application will take search criteria input by the user and return a list of locations on and around the user's respective campus. The system will query various data sets and API's to find the results closest to what the user is requesting. The minimum deliverable will include a search page that will return a generated list using real time datasets based on search criteria, an account page, and a home page.

The user interface of the application is aimed to deliver a minimal, yet modern design built on top of Angular and the Bootstrap framework. By leveraging these technologies, the interface will be fluid when moving from desktop to mobile, and from mobile to tablet. It will scale appropriately and be fully mobile responsive. The app will be built on top of service workers, which will allow the application to be a progressive web app, which allows it to work in offline mode just as it would when it is connected to wi-fi or mobile data. The user interface will be useable by people with disabilities as well. We will ensure the app is up to standard with the Americans with Disabilities requirements.

When the user logs in, they will be taken to a home page. This page will give a series of suggestions (coffee shops, restaurants, libraries, etc.), as well as places they had visited recently.

This will open up new places that the user may not have been aware of and may generate foot traffic for local businesses that are less popular. The profile page will be broken down into two subpages, which are the profile information (username, password, email address, etc.) and a reservation history page. This will not only let students see where they've been in the past, but also allow them to re-reserve a particular location if they regularly visit.

The middleware portion of Book It will leverage NodeJS and Express to create routes that will generate JavaScript Object Notation (JSON) data objects from the data stored on the MySQL databases. This will allow a constant, consistent state of data moving from the databases to the client side. The logic within the back-end will be responsible for parsing large amounts of data from many different sources for the middleware layer to consume it.

Project Schedule

Figure 3: Gantt Chart and Table 2: Work Breakdown Structure outline the projected schedule for completion.

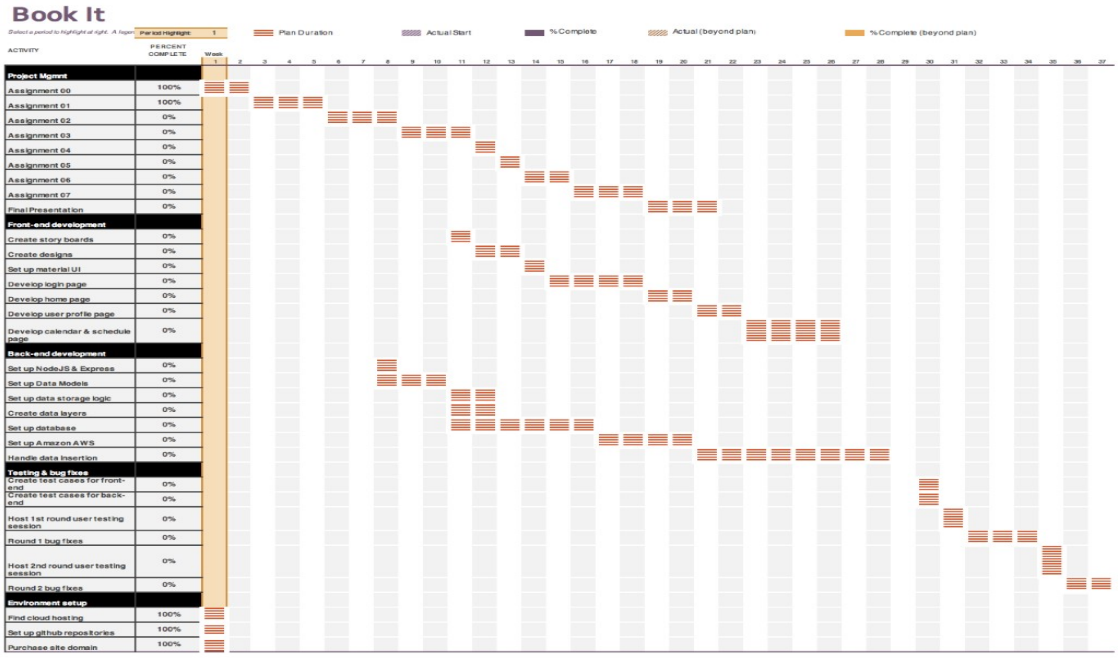

Figure 3: Gannt Chart

Task Name:	Start / End date:	Time (Days):
i. **Project Management**		
a. Assignment 0: Team members & project name	8/20/2018 – 9/03/2018	14
b. Assignment 1: Team contract	9/03/2018 – 9/24/2018	21
c. Assignment 2: Project abstract for tech. expo	9/24/2018 – 10/15/2018	21
d. Assignment 3: Team contract resubmission	9/24/2018 – 10/15/2018	21
e. Assignment 4: User Profile	10/15/2018 – 10/22/2018	7
f. Assignment 5: Use case diagram	10/15/2018 – 10/22/2018	7
g. Assignment 6: Draft report	10/22/2018 – 11/05/2018	14
h. Assignment 7: Final fall semester report	11/05/2018 – 11/26/2018	21
i. Final presentation	11/05/2018 – 11/26/2018	21
ii. **Front-end development**		
j. Develop story boards	11/01/2018 – 11/05/2018	5
k. Create designs	11/05/2018 – 11/07/2018	3
l. Set up Material UI	11/07/2018 – 11/10/2018	4
m. Develop login page	11/10/2018 – 11/23/2018	14

n. Develop home page	11/23/2018 – 12/06/2018	14
o. Develop user profile page	12/06/2018 – 12/20/2018	14
p. Develop calendar & scheduling pages	12/20/2018 -01/04/2019	14
iii. **Back-end development**		
q. Setup up NodeJS & Express	10/01/2018 – 10/02/2018	1
r. Set up data models & Connections required for successful transfer of data	10/02/2018-10/22/2018	21

s. Set up logic to handle successful storage of data	10/22/2018-12/22/2018	60
t. Create layers for business data	12/22/2018-01/22/2019	30
u. Set up database model with corresponding table names	10/01/2018-10/15/2018	14
v. Set up Amazon Web Services to handle storage of data	10/01/2018-10/15/2018	14
w. Handle insertion of data coming from the client-side application	12/15/2018-02/15/2019	60
iv. **Testing & bug fixes**		
x. Create test cases for front-end	2/10/2019 – 2/11/2019	1

y. Create test cases for back-end	2/11/2019 – 2/12/2019	1
z. Host 1st round user testing session	2/12/2019-2/13/2019	1
aa. Round 1 bug fixes	2/13/2019-3/06/2019	21
bb. Host 2nd round user testing session	3/06/2019 – 3/07/2019	1
cc. Round 2 bug fixes	3/07/2019 – 3/20/2019	13
dd. Deployment of web app	3/21/2019 – 4/10/2019	19
v. **Environment Setup**		
ee. Find cloud hosting infrastructure	9/05/2018 – 9/06/2018	2
ff. Set up GITHUB repos	9/06/2018 – 9/07/2018	1
gg. Purchase site domain from domain service	9/07/2018 – 9/08/2018	1

Table 2. Work Breakdown Structure (WBS)

TECHNICAL ELEMENTS

How well an application does is heavily reliant on how well the software is designed and built. If good decisions were made, the app will function as designed. The opposite is also true. Figure 4: Application Architecture displays the software design of the Book It application. A detailed explanation of Book Its network, application layer, and database design will follow.

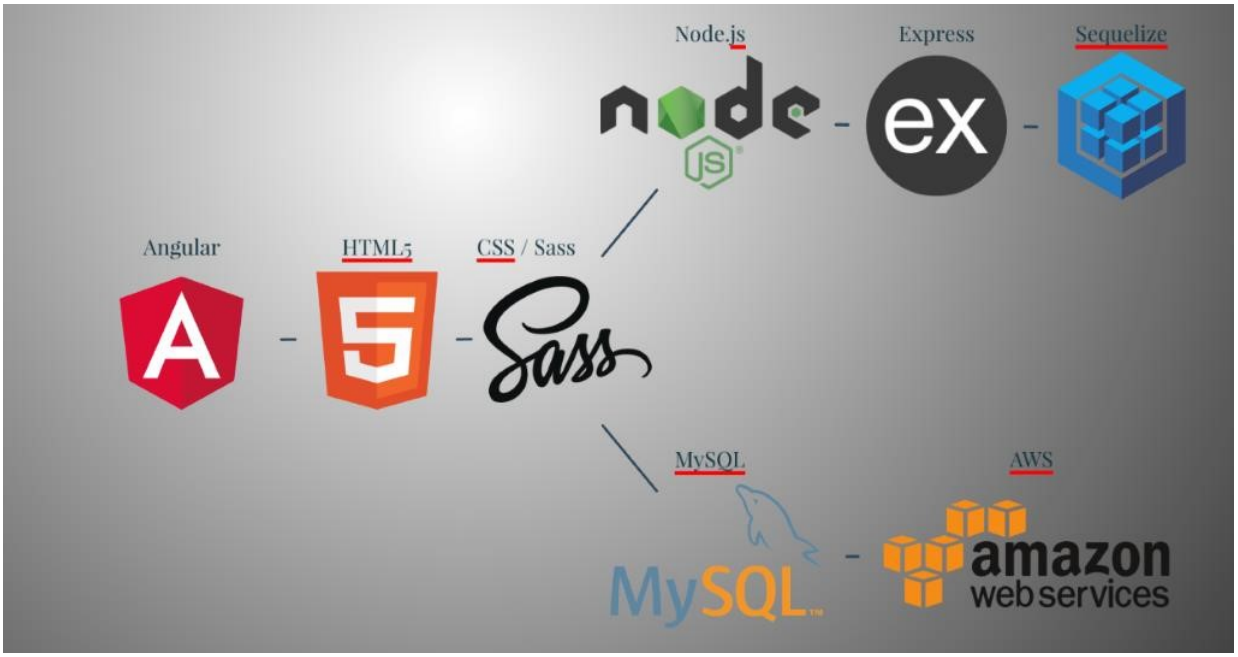

Figure 4: Application Architecture

Application

The applications back end will consist of NodeJS runtime environment and the Express web application framework, while the front-end will be built using the Angular library. Data will be transferred via JavaScript Object Notation (JSON) data objects between the front end and the back end.

Database

The application will utilize MySQL for the storage of data. Our MySQL database instance will be hosted in AWS to make our application easily scalable and less disaster prone.

MySQL was selected as our database of choice due to its track record of success in the industry. The data model will be built in a manner that allows it to be accessed by our tech stack. This will allow easy and quick transfer between all our layers.

Testing

For testing our app, we decided to go with an ad-hoc approach, as well as meeting twice in person to go over rounds of user testing. The usefulness for ad-hoc testing is to catch bugs as we are building the app, and not focus on a set schedule for testing. Since our team was so small, we decided ad-hoc testing was more important, as meeting our goals for expo trumped setting aside multiple specified times for testing. For example, when a developer would finish a feature, they would ad-hoc test that feature a minimum of 5 times before pushing the code to the GitHub repo. If, for whatever reason, a bug still made its way to our production ready code, the bug was logged in GitHub and assigned to the developer who oversaw that respective area. For example, if a bug related to the user interface surface, it would be logged and assigned to the user interface developer to fix. This testing process allowed for a smoother flow of work and led to reaching our completion date.

Test Plan

Overview

This document's purpose is to explain the methodology we will be implementing for Book-It and the testing process. The web application can run on any device with a browser installed. The individuals listed below should use this document as guide for the testing process.

- Developers

- Project Management
- Testers

Objective

To verify that the functionality of the application works as intended, the team will perform end-to-end testing. The tests outlined below are designed to test all core functionality of the application. During the development process, the team will Ad-Hoc test the process to ensure the new features work as designed.

Scope

The scope of testing is designed to test both the client-side and the server side, which will encompass virtually 100% of the application and its intended functionality. The scope of the testing process will be based on what is considered the requirements of the application.

Testing

The testing process will be performed in two ways. The first will be ad-hoc testing, which is performed on the spot as features get implemented. The second will be a scheduled time slot in the middle of March to the beginning of April. Each core piece of functionality will receive its own test file, which will test its functionality in its entirety.

Documentation

When the specified user is running tests and a bug is found, they should navigate www.github.com and log the bug in the issues tab. They should then assign the issue to the

developer in charge of the specified area (Front end, Back end). This will ensure the developer in charge of that piece of code can fix the bug in a timely manner, and there will be documentation showing what needs to be done.

Process

The process for testing the application is as follows:

- Create a thorough list of situations to test around
    - Write test cases for each situation.
- Write up the documentation to support each test case. This will include the specific user, the goal for the test, and the steps to get to the end of the test.
- If a bug is found when running the test, the user is to log the bug in the GitHub issues tab.
- There are two areas where tests will be written:
    - Client-side testing - This will include testing:
        - Various screen sizes (Mobile, Tablet, and Desktop)
        - Various operating systems (Windows, iPhone, Android, Mac)
        - Ensure inputs, buttons, modals, etc. all work appropriately
        - Ensure there are no visual elements that have issues.
        - Ensure there are no typos, spacing issues, etc.
    - Back-end testing - This will include testing:
        - Ensuring the signup & login process works
        - Edit account data
        - Search for specified data

- Adding invitees to events
- Scheduling new events, and editing existing events

Determining Pass/Fail Conditions

For the application to be successful in today's market, the application must work as intended 100% of the time. Therefore, a successful test plan is crucial to its success. For this reason, all the tests must pass 100% of the time. If a test fails, the tester is expected to log the failing test in the GitHub issues tab, and assign the failing test to a developer, as well as documentation pertaining to the environment tested in.

Schedule

While Ad-Hoc testing will happen as new features are implemented, the core testing will be done in two one-week segments during the middle-to-end of March, once we move into code freeze. The first will test the client side from end-to-end, and the second round will test the server-side from end-to-end. This will leave roughly 1.5 weeks to fix any bugs that are found.

Type	Dates
Round 1 – Client-side	March 10,2019 – March 16, 2019
Round 2 – Server-side	March 17, 2019 – March 23, 2019
Round 3 – Bug Fixes	March 24, 2019 – April 4, 2019

What we learned

While we have not officially moved into our testing phase, we have been ad-hoc testing over the last semester and a half. This has taught us a few things. Testing as you implement features is crucial. There is nothing worse than building out a ton of new functionality, only to find out the very first thing that was built is critically affecting all the new code. Making small

commits to GitHub and tracking all known bugs in the issues tab help with the process of getting bugs fixed in a timely manner.

Testing Reports - AD-Hoc - 11/01/2018 - 03/09/2019

Tester	Date/Time	Desired	Actual	Pass/Fail	Bug?
Developer	Ad Hoc	User can enter text into search bar and results are returned, for both on campus and off campus locations.	Results are returned	**Pass**	**No**
Developer	Ad Hoc	User can login & logout to and from the web app.	User successfully logs in	**Pass**	**No**
Developer	Ad Hoc	App loads successfully on various screen sizes in chrome dev tools.	App is successful at loading correctly on mobile, table, and desktop	**Pass**	**No**
Developer	Ad Hoc	Test for object data coming back from the server side and showing in the desired location on the UI.	Data is returned successfully in the correct location	**Pass**	**No**
Developer	Ad Hoc	New user can sign up for the web app.	The user can enter credentials and successfully sign up to use Book-It.	**Pass**	**No**

Developer	Ad Hoc	User can successfully navigate to the correct page when they interact with a link.	User navigates to the intended page when clicking on a link.	**Pass**	**No**

Testing Reports – Client-side – March 10, 2019 – March 16, 2019

Tester	Date/Time	Title	Desired	Actual	Pass/Fail	Bug?
Jeremy	03/10	Routes	**Routes work**	**Routes work**	**Pass**	**No**
Jeremy	03/10	Buttons	**Buttons work**	**Buttons work**	**Pass**	**No**
Jeremy	03/13	Colors	**Ui is consistent**	**UI is consistent**	**Pass**	**No**
Jeremy	03/13	Login	**Login works**	**Able to login**	**Pass**	**No**
Jeremy	03/13	Logout	**Logout works**	**Able to logout**	**Pass**	**No**
Andy	03/16	Updating user information	**User can update their information**	**Updating works**	**Pass**	**No**
Andy	03/16	Scheduling	**User can schedule**	**Scheduling works**	**Pass**	**No**
Andy	03/16	Mobile responsive	**App works correctly on mobile**	**Mobile performs as expected**	**Pass**	**No**

Testing Reports – Server-side – March 17, 2019 – March 23, 2019

Tester	Date/Time	Title	Desired	Actual	Pass/Fail	Bug?
Deonte	03/17-03/22	**Routing in postman**	**Routes are working in postman**	**Routes work when checked in Postman**	**Pass**	**No**
Deonte	03/17-03/22	**Queries work**	**Queries return the correct data set**	**The correct data is returned**	**Pass**	**No**
Deonte	03/17	**API is functioning**	**The YELP API is working correctly**	**YELP returns correct data**	**Pass**	**No**

USER INTERFACE

History Page

Figure 5: History page, will allow the user to view their past reserved events. This page will feature a Navigation bar generated by Bootstrap, as well as a left side bar that will be populated with links that will allow the user to move between the ReactJS components. The middle section will be where the results are shown, and when the list is loading the user will be given a loading icon that presents a visual indicator, so they are not left confused to what is happening. This page will be accessible by all users.

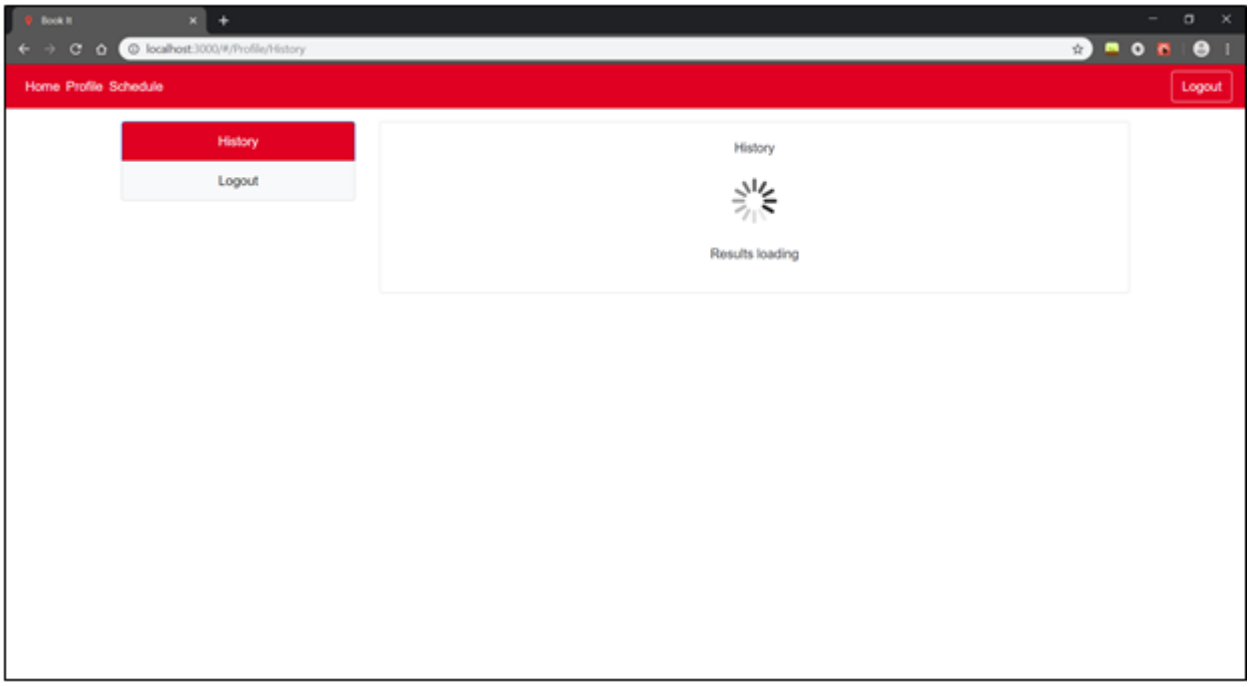

Figure 5: History Page

Profile Page

Figure 6: Profile page will allow the user to view their past reserved events. This page will feature a Navigation bar generated by Bootstrap, as well as a bar that will be populated with links that will allow the user to move between the Angular links. The middle section will be where the component (History) will be shown. This page will eventually be populated with more links, but for now we only have a history link.

Figure 6: Profile Page

Schedule Page

Figure 7: Schedule page will allow the user to search for a location. This page will feature a Navigation bar generate by Bootstrap. There is a search bar in the main component, where the user will be able to enter their desired search criteria, and the search bar will query the yelp API, as well as local databases for on-campus and off-campus results. The results will be populated

below the search bar, as well as a calendar, input fields for relevant information (name, other users, etc.), and submit button. This will allow the user to schedule events all on one page.

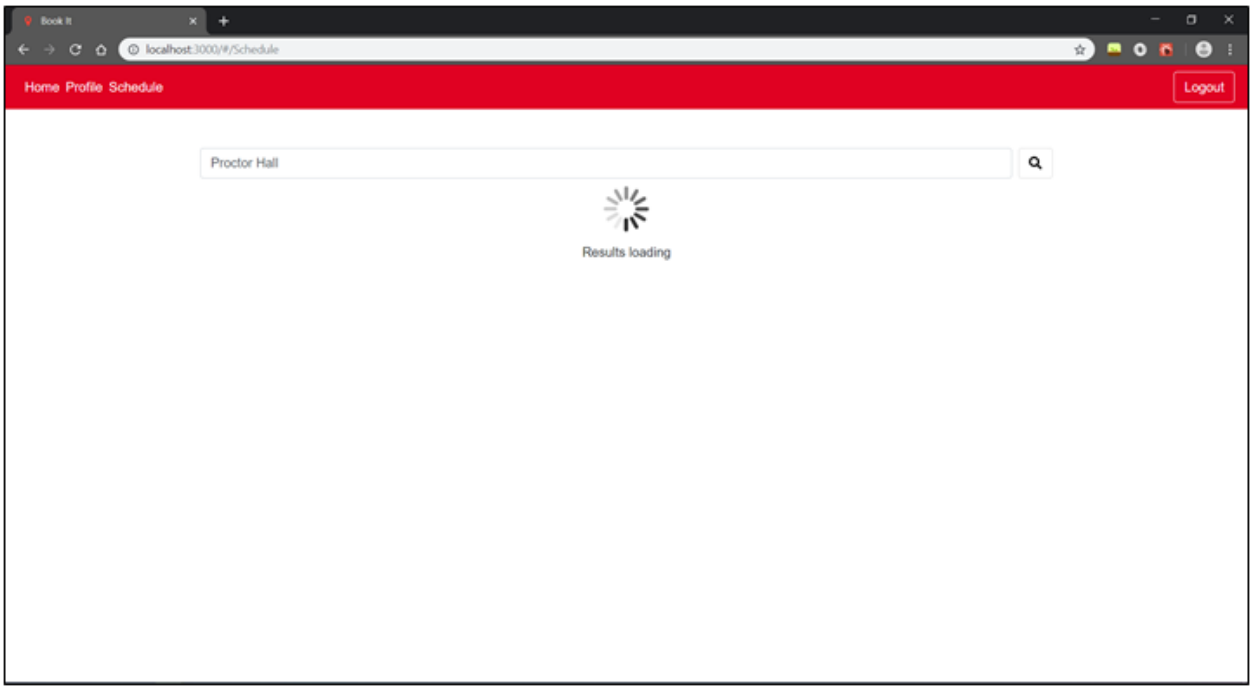

Figure 7: Schedule Page

CONCLUSION

Spring Semester

The spring semester went well overall, although we had a few challenges along the way. The first issue we ran into was staying on track. This caused a number of issues to stem up, including tasks taking longer than expected, and team members not always knowing what was happening. The other primary issue was running into issues with the APIs we utilized for our data. This could have been avoided, had we read the documentation more thoroughly going into the semester.

The skills the team learned were all beneficial overall. The first, and most important, was how to work with other people. This is very important in the real world, and it is something all developers should know how to do. All the members of the team picked up some new technologies as well. Whether it was JavaScript, HTML & CSS, or Node & AWS, every person on the team walked away knowing more than they knew, which goes a long way in our respective fields. Lastly, knowing how to come up with time estimates, writing up requirement documentation, and learning when and how to alter scope is a crucial skill.

Going into the spring semester, we moved from React to Angular, but kept the rest of the app primarily the same. The reason for this move was technical knowledge. This allowed us to meet most of the goals we needed to on time, outside of 1 or 2 things not being wrapped up by expo. In the spring, we rebuilt the UI completely, built all the remaining logic out using Angular, and moved everything to AWS. We had roughly 50% finished going into the second semester, and we wrapped up the remaining 50% a few weeks prior to expo.

At Expo, the team learned how to showcase a working model of a technology. There is never a shortage of questions for a team when new people view your application and learning how to explain this to people of all technical skill levels is a very important talent to have. Our team was able to answer extremely technical questions, as well as explain the benefits and importance our application could have on the world, if it were made production ready.

Fall Semester

While creating this application we faced our fair share of challenges. Whenever a team builds a product from the ground up, there are a fair share of hurdles that must be overcome. During the initial brainstorming process, we had a dozen or so ideas. We quickly narrowed the dozen ideas down to a few, and after some discussion we decided Book It was going to solve an issue that all three of us had faced over the course of our time at UC. We concluded that there is currently a solution that exists. Currently at UC, there is not a single solution to search the campus to find a place to study. Students must utilize any number of apps to search for a location, reserve it, and let their team members know. Book It aims to fix this issue. Our end goal is to alleviate these pain points for students and make the process easier.

After we narrowed down the overall idea, we had several more chokepoints we had to overcome – application name, tech. stack, scope of our project, and finding open source & free to use libraries and technologies. Since we only have roughly 9 months to build this application from the ground up, we came up with a.) initial scope and b.) backlogged ideas for after we completed our initial scope.

Up to now, we have been making solid progress with the creation of our application. We utilize Slack for communication, Google Drive houses our documents, and we utilize GitHub to

host our codebase. We signed up for the free tier of AWS, and that houses our MySQL database and we plan on hosting our live code on an instance of AWS. We use open source JavaScript libraries and frameworks to build our application.

We are currently working on building out our ReactJS codebase that will render the front end, and we will couple ReactJS with React Redux to handle ensuring the data we are receiving and returning from our middleware is in a consistent state. Our middleware is handled by NodeJS and Express, and the routes are currently being expanded upon to retrieve the data from the MySQL database. Our Database is currently in development, and the tables are in the process of being built out and normalized.

Future Recommendations If this project were to be redone, a more definite scope would have been put into place. This would have made things go more smoothly. Along the course of the two semesters, our team moved tasks from the current scope to the wish list and back. This made it harder to stay on track, and things didn't get done as quickly as they should have. Making sure everyone always had a task to work on would have been changed as well. Along the way, it felt like some individuals had more work than others at any given time. This led to frustration that was unneeded and should not have happened.

If more time were given, more social elements would have been implemented into place, as well as an admin panel. These were two major elements we should have finished, but because of time constraints these elements were never finalized. This led to pieces of the web-app that were crucial to its final form missing. We would have also put more polish and shine on the User Interface and User Experience. While the application looked good and functioned well in its final form, it could have looked and functioned even better.

To others who are getting ready to take this course, it is highly recommended that you find individuals who play off your strengths and weaknesses when picking team members. This will allow for a better flow as a team, and all aspects of the application will be handled. Utilize as many open source tools as possible and update them regularly. Our team used Slack, Google Hangouts, Trello, just to name a few. This will cut down on costs significantly and allows for better team collaboration.

Currently, there are no further plans for this project as the team stands. If one member wants to take this application and take it further, the other members are happy to let that member move forward.

Poster

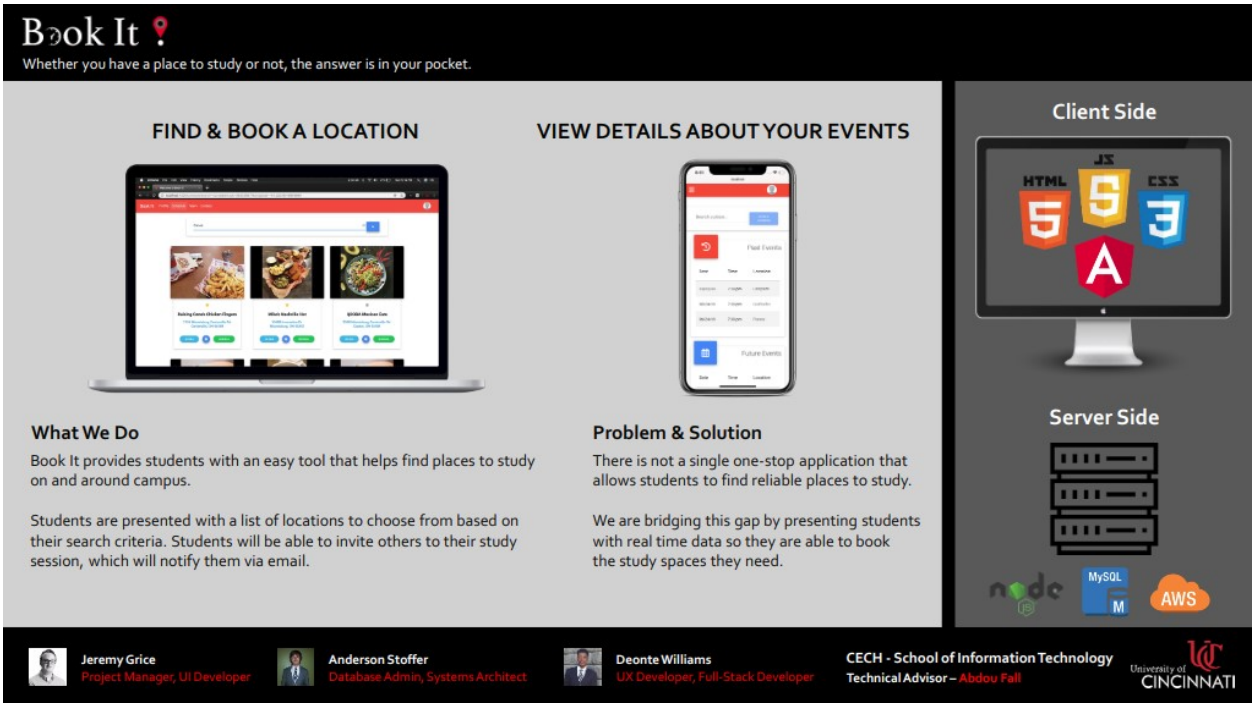

Figure 8: Poster

APPENDIX B. REFERENCES

"How College Students Spend Their Time" Center for higher Education Enterprise - May 11, 2016. https://chee.osu.edu/news-events/2016/05/test-post-4/