UC CTF

by

Taylor Reilly, Sydney Gleckler, Devaughn Penn

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

Taylor Reilly	**4/26/20**
Taylor Reilly	4/26/20
Sydney Gleckler	**4/26/20**
Sydney Gleckler	4/26/20
Devaughn Penn	**4/26/20**
Devaughn Penn	4/26/20
Ryan Moore	**4/26/20**
Ryan Moore, Technical Advisor	4/26/20

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

April 2020

Table of Contents

List of Illustrations

Tables

Figures

Abstract

According to cybersecurityventures.com, despite the high demand for new talent in the cyber security field, there are currently not enough cyber professionals to fill this need (Morgan, 2019, p. 1). To generate new cyber talent, the Ohio Cyber Range Institute utilizes its in-house web application to host regional Capture-the-Flag (CTF) events—exciting, cyber-themed competitions that help inform and spur interest in cyber security. Our project, UC CTF, takes that existing web application and greatly expands upon it. UC CTF features a completely redesigned application site, enhanced functionality and quality-of-life improvements for site administrators, and a brand new question repository of fun cyber challenges for CTF players—from beginner to expert and everything in between. UC CTF greatly improves upon the capabilities of the original application, allows for players of all skill levels to engage in an exciting CTF environment, and helps to develop new talent in cybersecurity.

1. INTRODUCTION

1.1 Problem

According to CyberCrime Magazine, there will be 3.5 million unfilled cyber security positions by 2021 (Morgan, 2019, p. 1). Additionally, with an estimated global shortage of 2 million cyber security professionals in 2019 (p. 1), the need for cyber security on a global scale is becoming critical. Despite this need, there are currently very few educational solutions that bolster interest and help generate new talent in the cyber security industry. According AT&T Cybersecurity, this is causing a "shortage of talent" (Brown, 2019, p. 1), which could spell disaster for the organizations desperately in need of cyber security.

So, how is this critical need met? According to the website *Lastline*, "One activity that can help build a team, and at the same time improve the security skills of the people involved, is participating in Capture the Flag (CTF) competitions" (Vigna, 2019, p.1). CTFs are, essentially, events that host cyber security challenges for players in a fun, competitive environment. Organizations such as DEFCON, Google, and Facebook host their own in-house CTF events; however, due to their highly competitive nature, going to such events may be intimidating for beginners, or for those with only a recent interest in cyber security. Current CTF applications tend to be targeted towards specific skill levels, especially experts and those with advanced skills, while neglecting to cater to all types of skill levels. Currently there are no CTF applications that encourage a wide range of all different skill levels—from experienced professionals to beginners—to play in the same organized event environment.

The Ohio Cyber Range Institute (OCR) has its own in-house Capture-the-Flag web application. Their existing app, however, has a limited number of challenge exercises, and the current build of its software architecture is incomplete and missing many features. For the CTF events hosted by OCR's web application, it can make those CTF events seem similar or redundant, due to the utilization of the same cyber challenges and a general lack of features on the web application's end.

1.2 Solution

UC CTF is our team's effort to make the Ohio Cyber Range Institute's Capture-the-Flag web application a complete, robust app. The new and improved web application features a large number of challenges that cater to all sorts of player skill levels, from beginner to expert and everything in between, allowing for all competitors to coexist in an enjoyable CTF environment. These improvements to the software application allow players of all skill levels to have fun in a competitive environment, learn more about cyber security, and help to generate a greater interest in the cyber security industry. Our project is different from the other few CTF applications on the market because it features more challenges for participants while also accommodating skill levels of all types. Generally other CTF applications do not cater to beginner skillsets through expert.

1.3 Project Goals/Brief Methodology

Our goals included building upon the existing CTF software application owned by the Ohio Cyber Range Institute, adding substantial quality-of-life improvements to the

application, and creating a large repository of cyber security, digital forensic, steganography, cipher/decoding, and trivia-based CTF challenges for players of all skill levels.

1.4 Overview

The rest of this document describes our team's process of concepting the idea; project planning and methodology; source code development and functionality implementation; web application testing; and final reflections on the project as a whole.

2. DISCUSSION

2.1 Project Concept/Solution

The idea for a CTF software application came from Professor Moore, who approached us with this idea after our former idea, that of a wireless dog leash, had to be ended, due to conflicts surrounding animal testing with the University of Cincinnati's Animal Care and Use program. Professor Moore mentioned that this project would fit all of our team's individual academic IT tracks—that of Software Development, and Cyber Security. After determining which project idea would give us meaningful work, was unique, and would fit as a substantial Senior Design project, our team unanimously agreed on this Capture-the-Flag software application project.

2.2 Design Objectives

Our team decided on several design objectives for our project in the form of Project Goals. We discussed with our advisor at length which goals we should focus on, with the idea that all should primarily contribute to the overall success of the Ohio Cyber Range Institute's CTF web application.

2.2.1 Project Goals

Our team built upon the Ohio Cyber Range's existing Capture-the-Flag software application from its existing source code up, along with creating a large repository of new and exciting CTF challenges of various skills levels to accommodate all players. This allowed for the Ohio Cyber Range's CTF events to have a greater breadth of challenges

for participants of all skill levels. Adding quality of life features to improve the existing app also allowed for it to be much more engaging for CTF participants, making the software interface more intuitive for its site and event administrators.

After extensive discussions with Professor Moore, we decided on various goals for our project. General improvement features to the CTF web application include:

- A large repository of challenges for administrators of CTF events to pick and choose from, with differing challenge categories and skill levels.
- Improvements to administrator editing abilities for easier creation and management of teams, users, and challenge creators.
- CSS and HTML styling to create more responsive design for the web application and give it a new Ohio Cyber Range-themed color scheme.
- Completely new web app functionality for players and administrators alike, including but not limited to the ability to download all CTF data (players, teams, events, challenges), send “welcome” emails to new players, filter challenges by Category and Level, CTF players creating their own teams, and much more.
- Other quality of life changes to the software application such as improved player login and settings functionality, and a completely new homepage with additional links and pages.

All these features combined improve the software’s ease of use, making it a more comprehensive web application, and provide for it a larger repository of challenges. This all helps to improve the quality of CTF events that utilize the software.

2.2.2 Initial Goals That Had to be Abandoned

For Fall Semester, we determined a few stretch goals for our project in the event that we finished development earlier than anticipated and had extra time to do more development work. One of these goals was the implementation of a new real-time, graphical scoreboard for the web application's CTF events. Unfortunately, we had to abandon this goal for a few reasons. First, being a stretch goal, the scoreboard implementation wasn't a primary objective of our project, as it was something we would work towards only if we had the time to do so. Secondly, during our project development our team encountered a severe setback of several months (which is described more in-depth in Section 2.10, *Problems Encountered and Analysis of Problems Solved*, pgs. 34-36), which severely hindered our progress on development and meant we had to work on an extremely time-sensitive schedule. Finally, we discovered that the web application had an existing, fully implemented, completely functional real-time scoreboard already in its source code; because of this, we determined that "reinventing the wheel" and making a completely new scoreboard in this case was unnecessary.

Another stretch goal that we had to abandon was that of "Implementing a New Mode to Steal Points"; i.e., implementing a Red/Blue/Purple-teaming game mode for Capture-the-Flag events, where opposing teams could "hack" each other and steal points. After discussing this stretch goal at length with our advisor, we determined that such a game mode wouldn't really fit in with the idea of the web application's CTF events being for players of all skill levels. If more advanced players could just steal the points of less

experienced ones, it may be a discouraging experience for new players. For these reasons, this stretch goal was abandoned.

2.3 Methodology/Technical Approach

We achieved the goals we set for our project by following our design requirements and undergoing specific procedures for our development. For administrator and user functionality and capabilities, we drafted multiple mockup web pages that gave us an idea of how the application's web frontend will look. For general planning of the application, we created a flowchart diagram to illustrate the site's layout. For challenge ideas, we created an extensive list of them, coming to a total of over 60 all-new cyber-themed challenge questions for use at CTF events. Our challenges were a success as well; our advisor, Ryan Moore, informed us that our challenges were used for an actual Ohio Cyber Range CTF event, and that the administrators were more than pleased with the results and even plan on using them for future CTF events.

For the web application, after gaining access to its existing source code, our team spent time analyzing the source code, reading its documentation and comments, and familiarizing ourselves with the various programming languages utilized, including PHP, Typescript, Javascript, and the Yii Framework. We also set up Linux Ubuntu virtual machines (VMs) that allowed us to run the application in its current state, set it up on the VMs, and work on it from there as a team via the use of Integrated Development Environments (IDEs) such as Visual Studio Code.

We worked hard in our development phase, meeting the requirements and goals we set out to accomplish by following our planned development procedures to code and implement completely new functionality into the web application and its source code. At the conclusion of our development, we fully tested the web application through the use of several real-world software testing methodologies, meeting great success on that front. Following our design requirements we created for Fall Semester as a foundation for our project aided us tremendously in achieving our goals for our development and testing procedures in Spring Semester.

2.4 User Profile

Figure 1: User Profile - User offers a summary of one of two primary users of our project: the Users, or player in the CTF events.

USER PROFILE: **Capture- the- Flag Participant (User)**

PROJECT:
Capture- the- Flag Application

POTENTIAL USERS:
- First-time CTF participants
- Intermediate CTF participants
- Expert CTF participants

SOFTWARE, INTERFACE, AND RELATED EXPERIENCE:
This project will cater to CTF challenge beginners through CTF challenge experts. These individuals will be in the cybersecurity sector and have experience with either cybersecurity principles or white/grey/black-hat hacking.

First-time CTF participants will be of little-to -no experienced in these challenges and hacking abilities but will have some understandings of cybersecurity principles. They will still be able to utilize the CTF and be able to select challenges based on their skill range.

Regardless of skill level, CTF participants will be able to compete at a CTF event with challenges that not only include their skillset and also challenge them as well.

EXPERIENCE WITH SIMILAR APPLICATIONS:
Experienced CTF participants will have experience with other CTF challenge events and CTF principles. Our CTF application will give them the ability to choose the difficulty of their challenge with more categories and more challenges. Our CTF application will also better show how their performance and scores stacks up against other participants.

First-time CTF event will not have experience with an application like ours but may be knowledgeable of techniques used in our CTF challenges. They will be able to catch on with the ways of CTF through our help and tips and beginner-level challenges.

TASK EXPERIENCE:
Experienced CTF participants will have familiarity with our types of CTF challenges including but not limited to: steganography, cybersecurity Trivia, cryptography, binary analysis, reverse engineering, and mobile security & system.

FREQUENCY OF USE:
The CTF application will only need to be set up once. But CTF participants can use this application for as many challenges they participate in.

KEY PROJECT DESIGN REQUIREMENTS THAT THE PROFILE SUGGESTS:

- Tips and help on the nature of the CTF event
- CTF challenges based on difficulty level or points
- Account creation and login for new or existing accounts
- Easy- navigate able application

Figure 1: User Profile - User

As per Figure 1, users of the CTF web application will find their functionality is mostly determined as players of Capture the Flag events, where they can solve challenges and compete against other players and users.

Figure 2: User Profile – Admin offers a summary of the second of two primary users of our project: the Admin, or CTF event facilitators.

USER PROFILE: **Capture- the- Flag Facilitator (Admin)**

PROJECT: Capture- the- Flag application
POTENTIAL USERS: - CTF challenge event host
SOFTWARE, INTERFACE, AND RELATED EXPERIENCE: This project will be primarily targeted towards host/ organizers of CTF events. These users typically have a good understanding of both CTF events and cybersecurity principles. These individuals are typically in the cybersecurity field. This software will ensure that CTF hosts will have easily- customizable CTF events made to the specifications they want.
EXPERIENCE WITH SIMILAR APPLICATIONS: Since admins of CTF events are hosting, they typically have had experience in these events. Our product is different from others because our project eases the challenge in starting a CTF event by adding more challenges to choose from, more customization options, and ease of access of user creations.
TASK EXPERIENCE: CTF admins may have previous experience setting up CTF events selecting challenges, choosing color schemes, choosing display types, and being admin of teams/users.
FREQUENCY OF USE: This project will be used for as many CTF events the Admin would like to host. Each event will be customizable or saved past events can be re-used.
KEY PROJECT DESIGN REQUIREMENTS THAT THE PROFILE SUGGESTS: - Easy event setup - Challenge selection - Color scheme selection - Account management

Figure 2 : User Profile - Admin

As per Figure 2, Administrators have elevated privileges and authentication requirements compared to users, including the ability to deploy actual CTF events, manage Users and teams, download CTF-related data, and functionality related to general administration of the web application.

2.5 Use Case Diagram

Figure 3: Use Case Diagram shows the capabilities and functionality available to the two main users of the application: Users, and Administrators.

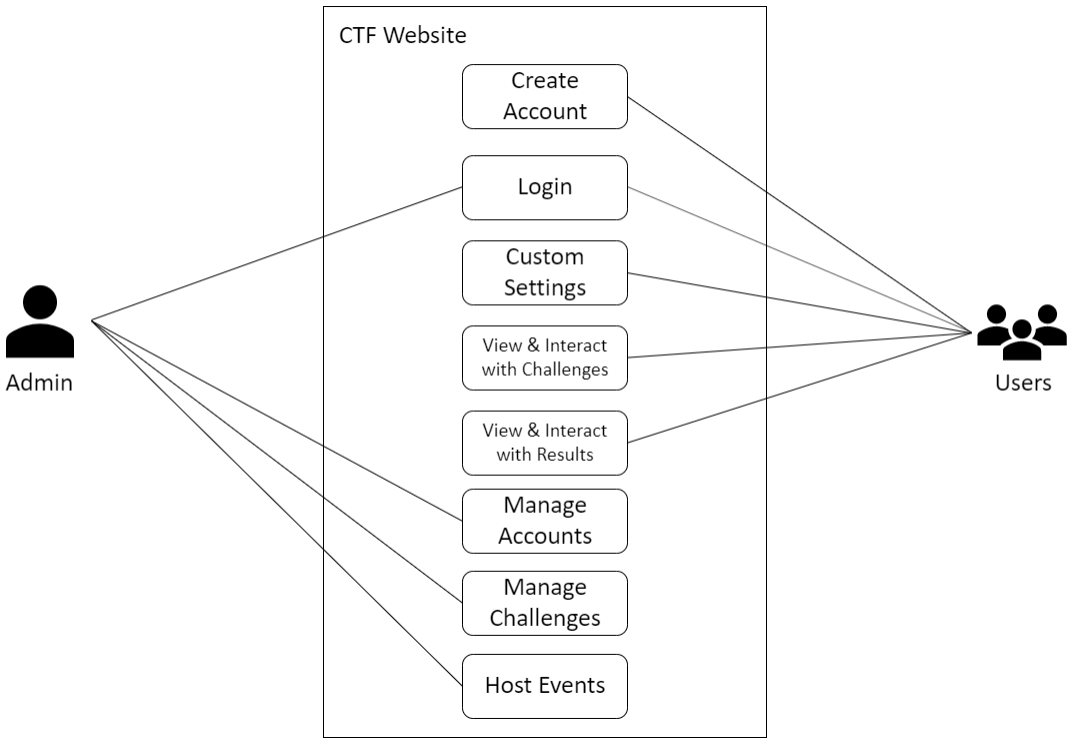


Figure 3 : Use Case Diagram

The Use Case Diagram helps as a visual comparison of the different roles that Admins and Users play, in regards to the administration and general usage of the CTF web application. Admins have more deeper-level privileges, while Users are surface-level.

2.6 Technical Discussion

The technical elements of our project cover three (3) primary areas: that of Networking, Application, and Security.

2.6.1 Network Overview and Discussion

Our team primarily utilized VMware Workstation as our network infrastructure to run the VMs for deploying our backend. More specifically, we utilized Linux Ubuntu VMs running Apache servers and MySQL databases. These VMs acted as our core backend servers for the purpose of running the web application. We used clients such as Git Bash to interact with these backend server VMs and push code changes to our project's Github repository.

2.6.2 Application Overview and Discussion

Our project utilized the programming languages that the existing web application already used, including Typescript, Javascript, and PHP. Furthermore, we utilized the application's existing MVC (Model-View-Controller) software architecture, plus the Yii backend framework, to build upon those implementations and code completely new functionality. We used IDEs such as Visual Studio Code to run the frontend of the application (while utilizing the backend servers in conjunction) to code it from there.

2.6.3 Security Elements Deployed

Because our project is that of a cyber security-oriented web application, it utilizes security concepts for its CTF challenge exercises for players to solve. The challenges feature cyber security topics such as Steganography, Computer Forensics, Decoding, and Cyber Trivia. The challenges can be solved by players via choosing from multiple choices, or by providing text-based answers.

Additionally, on the more technical side regarding security, the web application utilizes password login functionality and requires an admin login in order to access the app. Passwords are stored in a MySQL database and hashed, which protects the passwords from data breaches should the MySQL database be compromised. Additionally, the web application utilizes Angular authentication in the form of Auth Guards, interfaces attached to Typescript components throughout the source code that block unauthenticated users from accessing the various webpages and resources of the web application. Should a user attempt to access a part of the site they are not authenticated for, they are immediately redirected to the login page. This ensures that the site is only ever used by authenticated administrators, furthering the security of the application.

Finally, the Application Programming Interface (API) service that is utilized by the application only allows for usage by authenticated administrators. This means that the site is completely nonfunctional for non-authenticated users—meaning that the application's MySQL database and associated information is protected from non-authenticated, potentially malicious activity attempting to either insert, delete or modify sensitive data.

2.6.4 Technical Architecture Diagram

The following *Figure 4: Technical Architecture Diagram* is a visual depicting the technical layout of the web application, including its frontend, backend, and user/admin configuration.

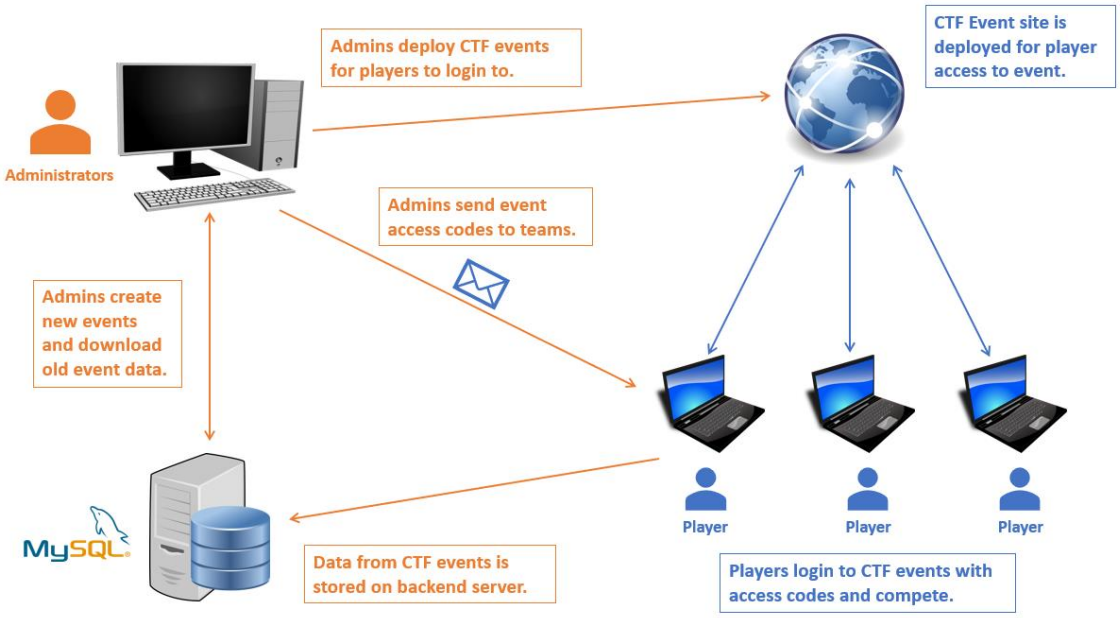


Figure 4: Technical Architecture Diagram

The Technical Architecture Diagram depicts how Administrators and Users interact with the web application and the specific roles they each play; namely, how Administrators facilitate the web application and its deployed CTF events, while Users act as the CTF players and interact with the web application on the User side.

2.6.5 User Interface

The following screenshot, *Figure 5: User Interface*, displays the interface of the CTF web application.

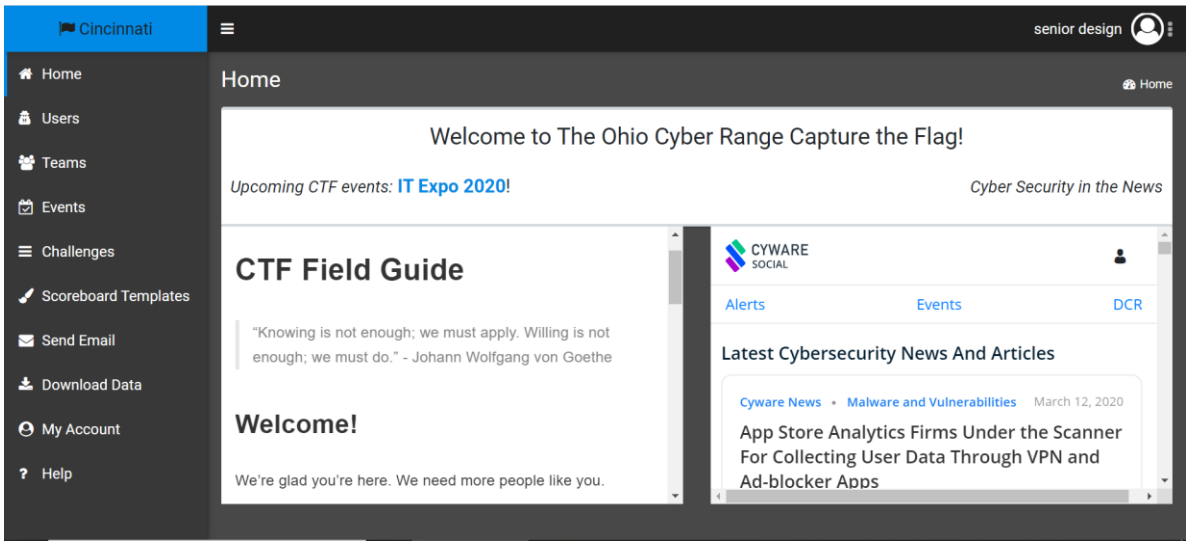


Figure 5: User Interface

In the upper-right-hand corner is the first and last name of the currently logged-in user. On the left-hand side is a vertical list of 10 pages that administrators can access. *Home* is the default landing page of the application; *Users* shows the current list of users for the application, pulling such data from the MySQL database; *Teams* shows the current list of teams and their associated users; *Events* shows a list of capture the flag events and their specific settings, such as their name, event date, etc.; *Challenges* is the repository of CTF questions, ranging from cyber-themed, to forensic, to trivia-based and more; *Scoreboard Templates* allows admins to choose a theme for a scoreboard for a given CTF event; *Send Email* allows admins to send a "welcome" email to players; *Download Data* allows admins to download all CTF data, such as from all users, teams, events, and challenges; *My Account* shows the currently-logged-in user's account settings and status; finally,

Help is a resource containing weblinks to the application's documentation, project Github repository, and the official Ohio Cyber Range Institute website and link to the OCR contact page.

Users, *Teams*, *Events*, *Challenges*, and *Scoreboard Templates* were existing pages for the application. For our project, we implemented the brand-new *Home*, *Send Email*, *Download Data*, *My Account*, and *Help* webpages, alongside a completely new color scheme and various quality-of-life improvements to existing functionality within the app.

2.7 Testing

For the Testing Phase of our project, we committed to several testing methodologies, objectives, and deliverables in order to ensure the overall success of our project.

2.7.1 Testing Overview and Methodology

In an overview of our testing plan, we utilized three testing methodologies:

- Development Testing (DEV): having the programmer of the code test their code.
- Quality Assurance Testing (QAT): having the team test the code to ensure quality.
- User Acceptance Testing (UAT): having the primary user of the application test the application as a whole. In our case, our advisor Ryan Moore is the User.

We chose this approach because this type of testing closely resembles real-world software development testing cycles. We used Microsoft Excel to log our test cases and keep track of our overall testing plan.

For our first Testing Methodology, Development Testing (DEV), the following steps were outlined and followed:

1. The programmer conducts a DEV test of their new code to ensure intended functionality.
2. If the DEV Test fails,
    a. the programmer troubleshoots their erroneous code and attempts to fix their code's issue(s).
    b. The programmer returns to step 1 of the DEV test methodology.
3. If the DEV Test is successful,
    a. the programmer Stages, Commits, and Pushes their code to the team's project UC CTF Github repository (with a detailed Commit message explaining their code).
    b. The programmer adds their changes to the team's Testing Plan sheet, changing the specific deliverable's State from "Defined", to "Ready".
    c. The Test Case deliverable will be moved up to Quality Assurance Testing.
4. Now the deliverable in question can continue forward to Quality Assurance Testing.

For our second Testing Methodology, Quality Assurance Testing (QAT), the following steps were outlined and followed:

5. Everyone on the team tests the deliverable to ensure intended functionality.
6. If the QAT Test fails,

   a. The QAT testers will mark the Test Case deliverable as a Bug.
   b. The QAT testers (including the initial creator of the Test Case) will Debug the code.
   c. Depending on the severity of the Bug, the initial Github commit may need to be rolled back until Debugging is resolved.
7. If the QAT test is successful (or if the Debugging has resolved successfully),
   a. The Test Case deliverable will be moved up to User Acceptance Testing.
8. Now the deliverable can continue to the final step of testing: User Acceptance Testing.

For our final Testing Methodology, User Acceptance Testing (UAT), the following steps were outlined and followed:

9. Once all Test Case deliverables of the project have moved up to User Acceptance Testing, the UAT test can begin.
10. The End User of the application tests the application for all intended and expected functionality of the application.
11. If the UAT Test fails,
    a. The UAT tester will report to the team of the failed UAT test and of what specifically failed.
    b. The team will mark the failed Test Case deliverable(s) as Bug(s).
    c. The team will Debug all deficiencies and code that the UAT tester reported.

d. Once the Bug(s) are fixed, the testing plan is restarted from DEV testing through the UAT Test.

12. If the UAT Test is successful,

    a. The UAT tester will inform the team of the successful UAT test.

13. The application and its required functionality are now fully tested.

2.7.2 Scope of Testing

Our team tested all newly implemented code and functionality that we implemented to this project, plus existing functionality to determine that our changes didn't negatively impact any existing code. We tested our project on both the User (CTF players), and Admin (CTF event administrators) side. To demonstrate this, the following diagram outlines all of the testing deliverables of the project.

Figure 6: Test Cases depicts a portion of the compiled list of specific test cases for our project testing phase.

Test Plans

ID	Corresponding Test Case	Test Case Title	User/Admin	State
1	Interact with Results	View scores/ challenge results through scoreboard	User	Ready
2	Interact with Results	Administrators can Download all site Capture the Flag data	Admin	Defined
3	Manage Accounts	Change Password Length from Standard 10 char to 8	Admin	Ready
4	Manage Challenges	Sort Challenges by Difficulty Level	Admin	Ready
5	Manage Challenges	Sort Challenges by Points	Admin	Ready
6	Manage Challenges	Filter Challenges by Difficulty Level	Admin	Defined
7	Manage Challenges	Filter Challenges by Points	Admin	Defined
8	Manage Challenges	Filter Challenges by Category	Admin	Defined
9	Host Events	Send Custom Email to Users	Admin	Defined
10	Host Events	Send "Welcome" Email to New Users	Admin	Defined
11	Create Account	Create account without supplying an email	Admin	Ready
12	Create Account	Create account password with 8 chars instead of 10	Admin	Ready
13	Create Account	Teams can create their own accounts	User	Defined
14	Login	Users can access team sign-up page	User	Ready
15	Login	Admins can sign into the application	Admin	Ready
16	Login	Admins can view New Homepage	Admin	Ready
17	Login	Admins can view new Send Email webpage	Admin	Ready
18	Login	Admins can view New Download Data webpage	Admin	Ready
19	Login	Admins can view new My Account webpage	Admin	Ready
20	Login	Admins can view new Help webpage	Admin	Ready
21	Custom Settings	Admins can choose new custom scoreboard for event	Admin	Defined
22	Custom Settings	Admins can change/save their custom account settings	Admin	Ready
23	View/Interact with Challenges	Users can view new Challenges from the new Repository	User	Ready
24	View/Interact with Challenges	Users can interact with new Challenges from the new Repository	User	Ready

Figure 6: Test Cases

Throughout testing, Test Cases remained a living document; therefore, it was consistently updated to reflect the progress of the testing phase.

2.7.3 Objectives

Our team decided on five primary Objectives for our project's Testing Plan.

1. Passing All Test Methodologies: All Test Case Deliverables must satisfactorily undergo all three Testing Methodologies: Development Testing (DEV), Quality Assurance Testing (QAT), and User Acceptance Testing (UAT), by the end of the team's project Testing time period.
2. Bugs: Any and all Bugs found during DEV, QAT, and/or UAT, no matter how small, must undergo Debugging.
3. Debugging Time Frame: Debugging must conclude upon the end of our testing time period in order to adhere to our testing schedule.
4. Communication: All Testers (DEV, QAT, and UAT) must communicate the results of their finished Tests to the team in a quick and efficient manner (preferably within 24-48 hours).
5. Adhere to Overall Scheduled Time Frame: As per our team's WBS and Gantt Charts, our project's testing will begin in March (the end of our planned development), and conclude in April (before IT Expo).

2.7.4 Logging Test and Procedures

For our testing logging and procedures, we closely followed the three key methodologies outlined above: that of Development Testing (DEV), Quality Assurance Testing (QAT), and User Acceptance Testing (UAT). Any bugs found were immediately input into a documented report. The following figure, *Figure 7: Bug Report*, shows an example of the report's layout:

Bug Report				
ID	**Corresponding Test Case**	**Test Case Title**	**Detailed Description of Bug**	**User/Admin**
2	Interact with Results	istrators can download Event-specific C	those challenges do not have answers, the	Admin
15	Manage Challenges	Admins can Create New Challenges	eated/supplied for challenges. (This bug al	Admin
20	Create Account	Teams can Create their Own Accounts	n't work on Development side, but works c	User and Admin
20	Create Account	Teams can Create their Own Accounts	an only be created if an admin is already s	User

Figure 7: Bug Report

The Bug Report remained a helpful guide for our team in keeping track of outstanding issues and bugs within the project. Using the Bug Report alongside the Test Cases plan (see *Figure 6: Test Cases*, pgs. 20-21) was crucial to ensuring our team stayed on-task and on-schedule to fully completing our planned testing phase.

DEV testing remained the most straightforward; if a developer were working on their code and happened to find a bug within it, they would immediately work to rectify the issue. DEV testing was also the safest method of testing; a bug found during DEV testing only existed in that developer's specific code, meaning there was no danger of the bug affecting the actual application. Once the developer fixed all bug(s) and thoroughly tested their code, the developer would then Git Push their code to the project's code repository on Github.

For Quality Assurance Testing (QAT), the team tested the entire application as a whole. This is where we found a majority of the bugs in code. QAT testing was done informally, and testers would pick and choose test cases as they wanted. We did emphasize, however, that testers should ideally pick a test case they were not responsible for developing. This ensured that at least two different sets of eyes looked over every part of the code.

As an example of our team's QAT testing strategy, the following *Figure 8: Example QAT Test Case* is a screenshot showing the work of a tester actively testing the web app's Download Data functionality—in this case, downloading all CTF user and player data.

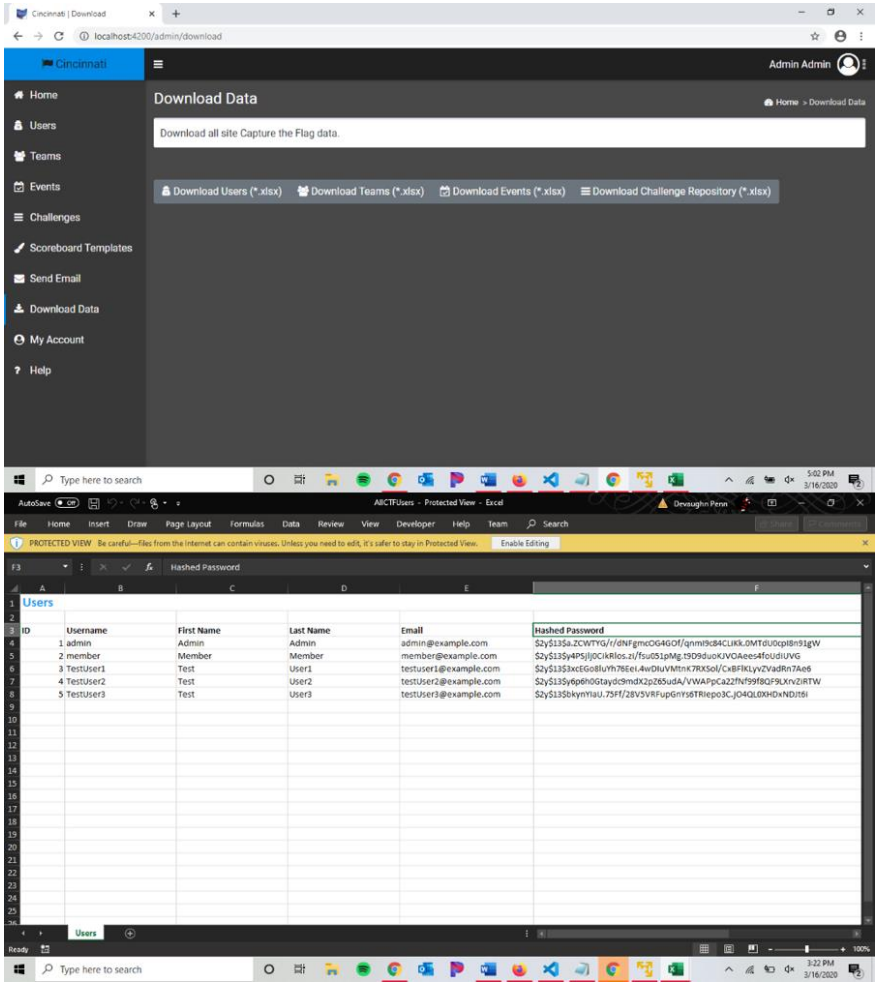


Figure 8: Example QAT Test Case

This QAT test was met with successful results, as usernames, first/last names, emails, and hashed passwords were, after pressing the “Download Users” button on the app site, generated in an Excel sheet as intended. Because of this specific test case’s success, it was marked as *Successful* in the Test Cases report’s *Results* column.

Finally, for User Acceptance Testing (UAT), the final stage of our testing, the team sent our project to our advisor, Ryan Moore, to get his technical feedback on the project, including final comments, questions, and critique.

Every test conducted—successful or unsuccessful—was logged appropriately in both our Test Cases and Bug Report documents, ensuring we were organized for the duration of our project’s Testing Phase.

2.7.5 Test Results

Our project’s utilized testing methodologies—DEV, QAT, and UAT—meant that our testing process was very Agile, relying on team/developer feedback and an overall sense of fluidity. This gave every tester on the team the freedom to test code in however way they were most comfortable with, with at least two objectives held: that of ensuring all code was bug-free, and that of completing the entire testing phase by April.

The following figure, ***Figure 9: Test Results***, shows our updated Test Cases report with a sample of the overall results of our project’s testing.

Testing Plan

ID	Corresponding Test Case	Test Case Title	User/Admin	State	Tester Name	Test Outcome
17	Host Events	Send "Welcome" Email to New Users	Admin	Ready	Devaughn	Successful
18	Create Account	Create account without supplying an email	Admin	Ready	Taylor	Successful
19	Create Account	Create account password with 8 chars instead of 10	Admin	Ready	Devaughn	Successful
20	Create Account	Teams can create their own accounts	User	Ready	Taylor	Successful
21	Create Account	Users can access team sign-up page	User	Ready	Taylor	Successful
22	Login	Teams can login to the Quiz Key page	User	Ready	Devaughn	Successful
23	Login	Admins can sign into the application	Admin	Ready	Taylor	Successful
24	Login	Admins can view New Homepage	Admin	Ready	Taylor	Successful
25	Login	Admins can view new Send Email webpage	Admin	Ready	Taylor	Successful
26	Login	Admins can view New Download Data webpage	Admin	Ready	Devaughn	Successful
27	Login	Admins can view new My Account webpage	Admin	Ready	Taylor	Successful
28	Login	Admins can view new Help webpage	Admin	Ready	Devaughn	Successful
29	Login	No Breadcrumbs appear on every webpage of the application	Admin	Ready	Taylor	Successful
30	Custom Settings	Admins can choose custom scoreboard for event	Admin	Ready	Devaughn	Successful
31	Custom Settings	Admins can change/save their custom account settings	Admin	Ready	Devaughn	Successful
32	Custom Settings	he My Account page loads the current logged-in user's setting	Admin	Ready	Taylor	Successful
33	View/Interact with Challenges	Users can view new Challenges from the new Repository	User	Ready	Devaughn	Successful
34	View/Interact with Challenges	sers can interact with new Challenges from the new Repositor	User	Ready	Devaughn	Successful

Figure 9: Test Results

By taking on test cases, documenting bugs as necessary, and debugging and rerunning tests until satisfactorily passed, we ensured that our testing phase went as smoothly and effectively as possible. As a result, at the end of our testing phase, all of our test cases passed 100% successfully.

2.7.6 What We Learned During Testing

During testing, we made an interesting discovery that the existing application already had several bugs within it. While this did add significantly to our testing workload, it was not surprising given the nature of our project was that of building onto an existing web application. It was also more difficult to debug these existing issues in the code because they weren't caused by *our* changes to the code, making these specific issues harder to pinpoint. In the latter case of a bug being on our end, simply going through the project's Git history can a lot of times help in pinpointing the exact issue of the bug itself. But when a bug is found that existed *before* our project's implementation, it definitely made debugging more difficult.

Overall, we learned that testing is an absolute crucial phase for any project. If project testing is neglected in any way, there are no guarantees that the project will work as intended, which can spell disaster at formal showings such as IT Expo. We also learned that DEV testing—developers testing alongside the building of their code—helps tremendously with the formal testing phase as well. Though DEV testing is done during the development phase, it absolutely helps to set a solid foundation for the actual, formal testing phase of a project, and in some cases, DEV testing can fix nearly all bugs before they become issues during QAT, or even UAT. Following all these testing methodologies was important to ensure the overall success of our project's presentation at IT Expo.

2.8 Budget

Table 1: Final Budget depicts our project's budget over the course of Fall and Spring Semester – the entire Senior Design Practicum.

Item	Months	Total Hours	Price Per Hour	Totals
Research	3 mo	120	$30 / hr	$3,600
Software Development	6 mo	360	$20 / hr	$7,200
Software Testing	2 mo	96	$15 / hr	$1,440
			Total Cost:	**$12,240**

Table 1: Final Budget

We estimated the average salaries of software development researchers, software developers, and software testers when pulling our data and numbers together and correlating it with our team's efforts on our project. Though our data shows an average total cost of $12,240, no actual money was spent on our project, meaning the actual cost of our project is $0.

The following *Table 2: Original Budget* is a depiction of the initial budget of our project.

Item	Months	Total Hours	Price Per Hour	Totals
Research	3 mo	120	$30 / hr	$3,600
Software Development	6 mo	200	$20 / hr	$4,000
Software Testing	2 mo	96	$15 / hr	$1,440
			Total Cost:	**$9,040**

Table 2: Original Budget

The clearest difference between our project's original and final budgets were the amount of hours spent on the project. We initially planned to spend an estimated 200 hours on our project's development phase; however, due to an unforeseen development troubleshooting issue of several months (see Section 2.10, *Problems Encountered and Analysis of Problems Solved*, pgs. 34-36), our development phase instead came closer to a total of around 360 hours, all team members' work combined.

2.9 WBS and Gantt Chart

Table 3: Work Breakdown Structure (WBS) is a depiction of our team's final, overall scheduling utilized for the completion of our project.

Task Name	Duration (Days)	Start Date	End Date
1.0 Project Management and Deliverables	248	8/26/19	4/29/20
1.1 Team Formation	4	8/26/19	8/29/19
1.2 Brainstorming	2	8/29/19	9/1/19
1.3 Fall Assignment 0	1	9/1/19	9/2/19
1.4 Fall Assignment 1	22	9/2/19	9/23/19
1.5 Fall Assignment 2	22	9/23/19	10/14/19
1.6 Fall Assignment 3	22	9/23/19	10/14/19

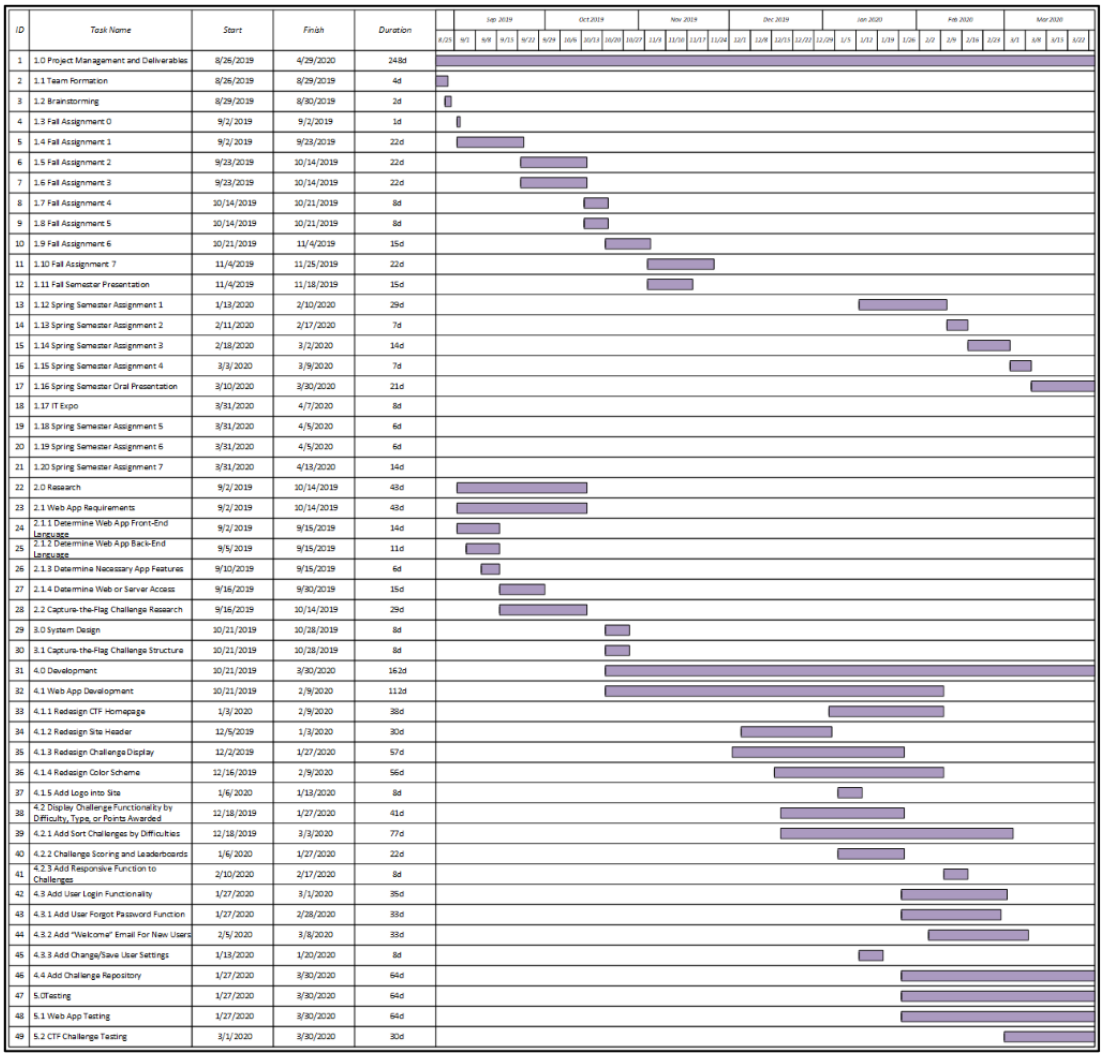
1.7 Fall Assignment 4	8	10/14/19	10/21/19
1.8 Fall Assignment 5	8	10/14/19	10/21/19
1.9 Fall Assignment 6	15	10/21/19	11/4/19
1.10 Fall Assignment 7	22	11/4/19	11/25/19
1.11 Fall Semester Presentation	15	11/4/19	11/18/19
1.12 Spring Semester Assignment 1	7	1/13/20	2/10/20
1.13 Spring Semester Assignment 2	7	2/11/20	2/17/20
1.14 Spring Semester Assignment 3	7	2/18/20	3/2/20
1.15 Spring Semester Assignment 4	7	3/3/20	3/9/20
1.16 Spring Semester Oral Presentation	7	3/10/20	3/30/20
1.17 IT Expo	14	4/12/20	4/19/19
1.18 Spring Semester Assignment 5	7	4/1/20	4/6/20
1.19 Spring Semester Assignment 6	7	4/1/20	4/6/20
1.20 Spring Semester Assignment 7	7	4/16/20	4/29/20
2.0 Research	42	9/2/19	10/14/19
2.1 Web App Requirements	42	9/2/19	10/14/19
2.1.1 Determine Web App Front-End Language	13	9/2/19	9/15/19
2.1.2 Determine Web App Back-End Language	10	9/5/19	9/15/19
2.1.3 Determine Necessary App Features	5	9/10/19	9/15/19
2.1.4 Determine Web or Server Access	14	9/16/19	9/30/19
2.2 Capture-the-Flag Challenge Research	28	9/16/19	10/14/19
3.0 System Design	8	10/21/19	10/28/19
3.1 CTF Challenge Structure	8	10/21/19	10/28/19
4.0 Development	162	10/21/19	3/30/20
4.1 Web App Development	112	10/21/19	2/9/20
4.1.1 Redesign CTF Homepage	38	2/9/20	1/13/20
4.1.2 Redesign Site Header	30	12/5/19	1/3/20
4.1.3 Redesign Challenge Display	57	12/2/19	1/27/20
4.1.4 Redesign Color Scheme	56	12/16/19	2/9/20
4.1.5 Add Logo into Site	8	1/6/20	1/13/20
4.2 Create Challenge Functionality w/ Difficulty, Type	41	12/18/19	1/27/20

4.2.1 Add Sort Challenge by Difficulty, Type, Points	77	12/18/19	3/3/20
4.2.2 Challenge Scoreboard for Scorekeeping	22	1/6/20	1/27/20
4.2.3 Add Responsive Function to Challenges	8	2/10/20	2/17/20
4.3 Add User Login Functionality	35	1/27/20	3/1/20
4.3.1 Add User Forgot Password Function	33	1/27/20	2/28/20
4.3.2 Add “Welcome” Email for New Users	33	2/5/20	3/8/20
4.3.3 Add Change/Save User Settings	8	1/13/20	1/20/20
4.4 Add to Challenge Repository	64	1/27/20	3/30/20
5.0 Testing	64	1/20/20	3/30/20
5.1 Web App Site Testing	64	1/27/20	3/30/20
5.2 CTF Challenge Testing	30	3/1/20	3/30/20

Table 3: Work Breakdown Structure (WBS)

Our Work Breakdown Structure has been our team’s primary deliverable scheduling mechanism that has ensured we are on time for completing our project.

Table 4: Gantt Chart is a more visually appealing depiction of our team’s final scheduling of deliverables for the project.



ID	Task Name	Start	Finish	Duration
1	1.0 Project Management and Deliverables	8/26/2019	4/29/2020	248d
2	1.1 Team Formation	8/26/2019	8/29/2019	4d
3	1.2 Brainstorming	8/29/2019	8/30/2019	2d
4	1.3 Fall Assignment 0	9/2/2019	9/2/2019	1d
5	1.4 Fall Assignment 1	9/2/2019	9/23/2019	22d
6	1.5 Fall Assignment 2	9/23/2019	10/14/2019	22d
7	1.6 Fall Assignment 3	9/23/2019	10/14/2019	22d
8	1.7 Fall Assignment 4	10/14/2019	10/21/2019	8d
9	1.8 Fall Assignment 5	10/14/2019	10/21/2019	8d
10	1.9 Fall Assignment 6	10/21/2019	11/4/2019	15d
11	1.10 Fall Assignment 7	11/4/2019	11/25/2019	22d
12	1.11 Fall Semester Presentation	11/4/2019	11/18/2019	15d
13	1.12 Spring Semester Assignment 1	1/13/2020	2/10/2020	29d
14	1.13 Spring Semester Assignment 2	2/11/2020	2/17/2020	7d
15	1.14 Spring Semester Assignment 3	2/18/2020	3/2/2020	14d
16	1.15 Spring Semester Assignment 4	3/3/2020	3/9/2020	7d
17	1.16 Spring Semester Oral Presentation	3/10/2020	3/30/2020	21d
18	1.17 IT Expo	3/31/2020	4/7/2020	8d
19	1.18 Spring Semester Assignment 5	3/31/2020	4/6/2020	6d
20	1.19 Spring Semester Assignment 6	3/31/2020	4/6/2020	6d
21	1.20 Spring Semester Assignment 7	3/31/2020	4/13/2020	14d
22	2.0 Research	9/2/2019	10/14/2019	48d
23	2.1 Web App Requirements	9/2/2019	10/14/2019	48d
24	2.1.1 Determine Web App Front-End Language	9/2/2019	9/15/2019	14d
25	2.1.2 Determine Web App Back-End language	9/5/2019	9/15/2019	11d
26	2.1.3 Determine Necessary App Features	9/10/2019	9/15/2019	6d
27	2.1.4 Determine Web or Server Access	9/16/2019	9/30/2019	15d
28	2.2 Capture-the-Flag Challenge Research	9/16/2019	10/14/2019	29d
29	3.0 System Design	10/21/2019	10/28/2019	8d
30	3.1 Capture-the-Flag Challenge Structure	10/21/2019	10/28/2019	8d
31	4.0 Development	10/21/2019	3/30/2020	162d
32	4.1 Web App Development	10/21/2019	2/9/2020	112d
33	4.1.1 Redesign CTF Homepage	1/3/2020	2/9/2020	38d
34	4.1.2 Redesign Site Header	12/5/2019	1/3/2020	30d
35	4.1.3 Redesign Challenge Display	12/2/2019	1/27/2020	57d
36	4.1.4 Redesign Color Scheme	12/16/2019	2/9/2020	56d
37	4.1.5 Add Logo into Site	1/6/2020	1/13/2020	8d
38	4.2 Display Challenge Functionality by Difficulty, Type, or Points Awarded	12/18/2019	1/27/2020	41d
39	4.2.1 Add Sort Challenges by Difficulties	12/18/2019	3/3/2020	77d
40	4.2.2 Challenge Scoring and Leaderboards	1/6/2020	1/27/2020	22d
41	4.2.3 Add Responsive Function to Challenges	2/10/2020	2/17/2020	8d
42	4.3 Add User Login Functionality	1/27/2020	3/1/2020	35d
43	4.3.1 Add User Forgot Password Function	1/27/2020	2/28/2020	33d
44	4.3.2 Add "Welcome" Email For New Users	2/5/2020	3/8/2020	33d
45	4.3.3 Add Change/Save User Settings	1/13/2020	1/20/2020	8d
46	4.4 Add Challenge Repository	1/27/2020	3/30/2020	64d
47	5.0Testing	1/27/2020	3/30/2020	64d
48	5.1 Web App Testing	1/27/2020	3/30/2020	64d
49	5.2 CTF Challenge Testing	3/1/2020	3/30/2020	30d

Table 4: Gantt Chart

Our team used the Gantt Chart, in conjunction with the Work Breakdown Structure, to ensure we remained on track for completing our project in a timely manner.

Table 5: Fall Semester Work Breakdown Structure (WBS) is a depiction of our team's initial scheduling of deliverables for our project during Fall Semester.

Task Name	Duration (Days)	Start Date	End Date

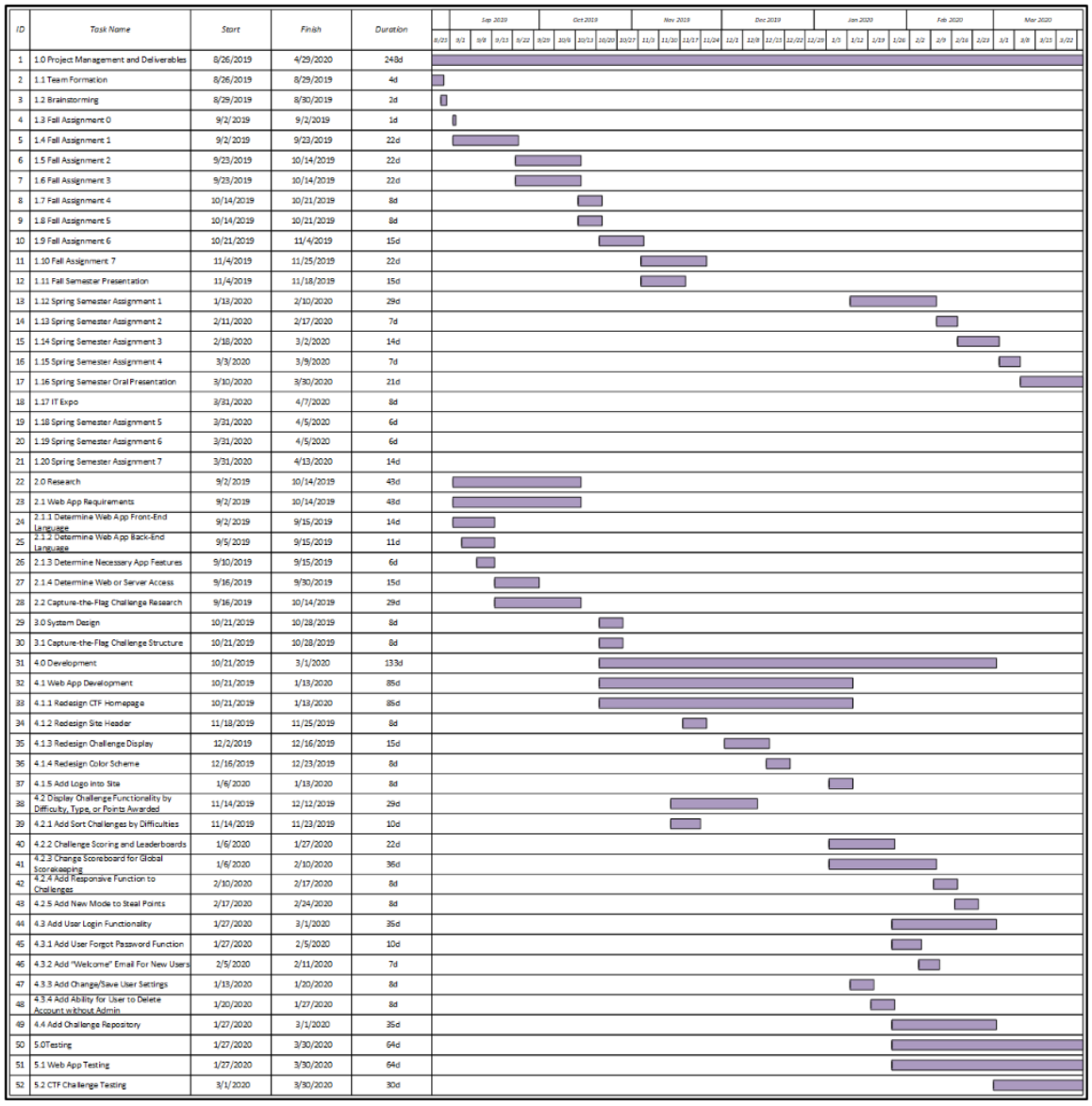
1.0 Project Management and Deliverables	248	8/26/19	4/29/20
1.1 Team Formation	4	8/26/19	8/29/19
1.2 Brainstorming	2	8/29/19	9/1/19
1.3 Fall Assignment 0	1	9/1/19	9/2/19
1.4 Fall Assignment 1	22	9/2/19	9/23/19
1.5 Fall Assignment 2	22	9/23/19	10/14/19
1.6 Fall Assignment 3	22	9/23/19	10/14/19
1.7 Fall Assignment 4	8	10/14/19	10/21/19
1.8 Fall Assignment 5	8	10/14/19	10/21/19
1.9 Fall Assignment 6	15	10/21/19	11/4/19
1.10 Fall Assignment 7	22	11/4/19	11/25/19
1.11 Fall Semester Presentation	15	11/4/19	11/18/19
1.12 Spring Semester Assignment 1	7	1/13/20	2/10/20
1.13 Spring Semester Assignment 2	7	2/11/20	2/17/20
1.14 Spring Semester Assignment 3	7	2/18/20	3/2/20
1.15 Spring Semester Assignment 4	7	3/3/20	3/9/20
1.16 Spring Semester Oral Presentation	7	3/10/20	3/30/20
1.17 IT Expo	14	4/12/20	4/19/19
1.18 Spring Semester Assignment 5	7	4/1/20	4/6/20
1.19 Spring Semester Assignment 6	7	4/1/20	4/6/20
1.20 Spring Semester Assignment 7	7	4/16/20	4/29/20
2.0 Research	42	9/2/19	10/14/19
2.1 Web App Requirements	42	9/2/19	10/14/19
2.1.1 Determine Web App Front-End Language	13	9/2/19	9/15/19
2.1.2 Determine Web App Back-End Language	10	9/5/19	9/15/19
2.1.3 Determine Necessary App Features	5	9/10/19	9/15/19
2.1.4 Determine Web or Server Access	14	9/16/19	9/30/19
2.2 Capture-the-Flag Challenge Research	28	9/16/19	10/14/19
3.0 System Design	8	10/21/19	10/28/19
3.1 CTF Challenge Structure	8	10/21/19	10/28/19
4.0 Development	133	10/21/19	3/1/20
4.1 Web App Development	85	10/21/19	1/13/20

4.1.1 Redesign CTF Homepage	85	10/21/19	1/13/20
4.1.2 Redesign Site Header	8	12/5/19	12/12/19
4.1.3 Redesign Challenge Display	15	12/2/19	12/16/19
4.1.4 Redesign Color Scheme	8	12/16/19	12/23/19
4.1.5 Add Logo into Site	8	1/6/20	1/13/20
4.2 Create Challenge Functionality w/ Difficulty, Type	58	12/18/19	2/13/20
4.2.1 Add Sort Challenge by Difficulty, Type, Points	58	12/18/19	2/13/20
4.2.2 Challenge Scoreboard for Scorekeeping	8	1/6/20	2/10/20
4.2.3 Add Responsive Function to Challenges	8	2/10/20	2/17/20
4.2.4 Add New Mode to Steal Points	8	2/17/20	2/24/20
4.3 Add User Login Functionality	35	1/27/20	3/1/20
4.3.1 Add User Forgot Password Function	10	1/27/20	2/5/20
4.3.2 Add “Welcome” Email for New Users	7	2/25/20	2/11/20
4.3.3 Add Change/Save User Settings	8	1/13/20	1/20/20
4.4 Add to Challenge Repository	35	1/27/20	3/1/20
5.0 Testing	70	1/20/20	3/30/20
5.1 Web App Site Testing	35	1/27/20	3/30/20
5.2 CTF Challenge Testing	30	3/1/20	3/30/20

Table 5: Fall Semester Work Breakdown Structure (WBS)

We edited this Fall Semester WBS and converted it to our new, final scheduling accordingly, which shows our most accurate progress and goals completed for the project.

Table 6: Fall Semester Gantt Chart is another depiction of our team’s initial Fall Semester scheduling.



ID	Task Name	Start	Finish	Duration
1	1.0 Project Management and Deliverables	8/26/2019	4/29/2020	248d
2	1.1 Team Formation	8/26/2019	8/29/2019	4d
3	1.2 Brainstorming	8/29/2019	8/30/2019	2d
4	1.3 Fall Assignment 0	9/2/2019	9/2/2019	1d
5	1.4 Fall Assignment 1	9/2/2019	9/23/2019	22d
6	1.5 Fall Assignment 2	9/23/2019	10/14/2019	22d
7	1.6 Fall Assignment 3	9/23/2019	10/14/2019	22d
8	1.7 Fall Assignment 4	10/14/2019	10/21/2019	8d
9	1.8 Fall Assignment 5	10/14/2019	10/21/2019	8d
10	1.9 Fall Assignment 6	10/21/2019	11/4/2019	15d
11	1.10 Fall Assignment 7	11/4/2019	11/25/2019	22d
12	1.11 Fall Semester Presentation	11/4/2019	11/18/2019	15d
13	1.12 Spring Semester Assignment 1	1/13/2020	2/10/2020	29d
14	1.13 Spring Semester Assignment 2	2/11/2020	2/17/2020	7d
15	1.14 Spring Semester Assignment 3	2/18/2020	3/2/2020	14d
16	1.15 Spring Semester Assignment 4	3/3/2020	3/9/2020	7d
17	1.16 Spring Semester Oral Presentation	3/10/2020	3/30/2020	21d
18	1.17 IT Expo	3/31/2020	4/7/2020	8d
19	1.18 Spring Semester Assignment 5	3/31/2020	4/5/2020	6d
20	1.19 Spring Semester Assignment 6	3/31/2020	4/5/2020	6d
21	1.20 Spring Semester Assignment 7	3/31/2020	4/13/2020	14d
22	2.0 Research	9/2/2019	10/14/2019	43d
23	2.1 Web App Requirements	9/2/2019	10/14/2019	43d
24	2.1.1 Determine Web App Front-End Language	9/2/2019	9/15/2019	14d
25	2.1.2 Determine Web App Back-End Language	9/5/2019	9/15/2019	11d
26	2.1.3 Determine Necessary App Features	9/10/2019	9/15/2019	6d
27	2.1.4 Determine Web or Server Access	9/16/2019	9/30/2019	15d
28	2.2 Capture-the-Flag Challenge Research	9/16/2019	10/14/2019	29d
29	3.0 System Design	10/21/2019	10/28/2019	8d
30	3.1 Capture-the-Flag Challenge Structure	10/21/2019	10/28/2019	8d
31	4.0 Development	10/21/2019	3/1/2020	133d
32	4.1 Web App Development	10/21/2019	1/13/2020	85d
33	4.1.1 Redesign CTF Homepage	10/21/2019	1/13/2020	85d
34	4.1.2 Redesign Site Header	11/18/2019	11/25/2019	8d
35	4.1.3 Redesign Challenge Display	12/2/2019	12/16/2019	15d
36	4.1.4 Redesign Color Scheme	12/16/2019	12/23/2019	8d
37	4.1.5 Add Logo into Site	1/6/2020	1/13/2020	8d
38	4.2 Display Challenge Functionality by Difficulty, Type, or Points Awarded	11/14/2019	12/12/2019	29d
39	4.2.1 Add Sort Challenges by Difficulties	11/14/2019	11/23/2019	10d
40	4.2.2 Challenge Scoring and Leaderboards	1/6/2020	1/27/2020	22d
41	4.2.3 Change Scoreboard for Global Scorekeeping	1/6/2020	2/10/2020	36d
42	4.2.4 Add Responsive Function to Challenges	2/10/2020	2/17/2020	8d
43	4.2.5 Add New Mode to Steal Points	2/17/2020	2/24/2020	8d
44	4.3 Add User Login Functionality	1/27/2020	3/1/2020	36d
45	4.3.1 Add User Forgot Password Function	1/27/2020	2/5/2020	10d
46	4.3.2 Add "Welcome" Email For New Users	2/5/2020	2/11/2020	7d
47	4.3.3 Add Change/Save User Settings	1/13/2020	1/20/2020	8d
48	4.3.4 Add Ability for User to Delete Account without Admin	1/20/2020	1/27/2020	8d
49	4.4 Add Challenge Repository	1/27/2020	3/1/2020	36d
50	5.0Testing	1/27/2020	3/30/2020	64d
51	5.1 Web App Testing	1/27/2020	3/30/2020	64d
52	5.2 CTF Challenge Testing	3/1/2020	3/30/2020	30d

Table 6: Fall Semester Gantt Chart

Our project's planning, development, and testing timelines remained crucially dependent in all versions of our team's Work Breakdown Structure and Gantt Charts. Ensuring these scheduling charts remained updated to reflect our most accurate progress on the project was a key factor in our project's overall success.

2.10 Problems Encountered and Analysis of Problems Solved

Throughout the project we experienced struggles with methods in which we would improve upon our project. Because the web app already exists we had to find the right cross between changing the framework while not hindering the system. This system was written in Typescript and PHP—languages our team was not initially proficient programming in—so our development was limited to what we learned as we developed. Another issue our team encountered pertained to customization. One main goal of our project was allowing CTF admins to customize their CTF challenge as much as possible, but we discovered that presenting admins with too many options can clutter the site, or even confuse admins. Our group struggled with finding the right combination of customization and simplicity.

To solve these issues, our group made sure to mockup our site with all options we wanted to prompt our admins with, and fully tested our site to ensure we would not confuse our users with too many options. We also took away some of our options we wanted to present our users that we felt may be unnecessary, such as manual account deletion for CTF participants.

Additionally, another problem we encountered was that of initially not being able to connect to the web application at all, time-wise setting us several months back in development. We struggled with setting up the existing application’s backend and frontend environments because its supporting documentation was very vague and missing crucial instructions, and its technologies were unfamiliar to us and required extensive

trial and error to get working properly. And, because we initially couldn't set up the application nor connect to its backend server API, it meant that we couldn't test any of the actual functionality of the project for the duration of our troubleshooting, such as adding/deleting users, creating events/new challenges, etc. All of the project's functionality interacted entirely with the API and its associated database—both of which were inaccessible to us. This meant we were unable to implement any new functionality for the duration of our troubleshooting the issue. Despite this setback we were still able to complete some surface-level development over the course of troubleshooting (though it required us to destroying every Auth Guard in the source code that blocked us from accessing the site without a login), such as implementing CSS color schemes, adding new webpages to the site navigation, and updating the site headers.

After several months of solid dedicated troubleshooting, however, our team was finally able to figure out the issue with the project—an erroneous php pointer within the source code, which was throwing hidden connection errors—and we fixed it accordingly. With that problem solved, we started our project's development immediately. Because this troubleshooting issue caused us to be behind schedule, this meant we essentially faced a crunch time of development and had to very quickly make up for lost time. While the crunch itself was not ideal, we still managed to complete all of our planned deliverables just before the start of our scheduled March testing phase, ensuring we were on schedule for our project. Despite setback, we worked hard as a team to make up for lost time, ensuring we completed all of our project development as planned, and we met our development goals with 100% success.

2.11 Future Recommendations

If our team had to do our project all over again, it would mean we would already understand the existing architecture, software, and programming languages used by the existing application. Additionally, we would already know how to set up the project and would not face any difficulties or lengthy troubleshooting with the setup phase like we did for our project. We would also already have our existing project planning and scheduling already written out. All of these things would give us an incredible advantage and head-start on software development for the project. If our team had to do this project all over again, with all these things in mind, we would certainly begin the software development phase immediately, and perhaps be able to accomplish even more than we already did for our project.

If our team had more time to work on our project, we would continue development on the project and possibly work towards implementing our project's stretch goals we had discussed. We would also discuss with our advisor even more new changes that he would like to see implemented into the web application, and work towards implementing those ideas and that functionality as well.

Regarding suggestions from others, we have had multiple ideas both regarding our project and just general tips for succeeding in the course. One such idea was that of branding—as the web app is the property of the Ohio Cyber Range Institute, we were suggested to avoid implementing any sort of University of Cincinnati branding to the web

application, in order to fully cement it as the Ohio Cyber Range Institute's web application. Another suggestion was that of communication. Communicating both with our professors and advisor remained a crucial component to succeeding in the course. We took this advice seriously and made an effort to keep a constant and regular contact with all our professors and advisors.

As our project—that of building onto an existing web application—is the property of the Ohio Cyber Range Institute, we will be relinquishing all of our newly implemented source code, Github project repository, and resource documentation to the Ohio Cyber Range Institute upon the conclusion of our project. As we will not be going forward with building onto this web application, we ensured that what functionality we *did* implement was properly commented, documented, and as clear as possible for future developers who will take on the work of continuing to build onto the Ohio Cyber Range's CTF web application.

3. CONCLUSION

3.1 Fall Semester Conclusion

Fall Semester was an incredible learning experience for our entire team. Throughout Fall Semester we were able to set a solid foundation for the groundwork of our project in the form of project planning and design, ensuring our overall success throughout the entirety of the Senior Design practicum.

3.1.1 Lessons Learned

For Fall Semester, our team's Lessons Learned were in the form of two facets: that of Communication, and Consistency. For *communication*, we learned that keeping a consistent dialogue with our teammates, advisor, and professor was crucial to staying on top of assignments and completing all our project deliverables. Foregoing communication for even a small period of time often led to disorganized results for our team, meaning communication proved an absolute necessity. This in turn led to our team learning the lesson of *consistency*—that of staying on top of assignments and producing well-written, complete and satisfactory work to the best of our ability, ensuring our team ended Fall Semester with high grades. Keeping both of these lessons in mind led to us completing Fall Semester with a solid foundational plan for the development of our project in Spring Semester.

3.1.2 Abilities/Skills Developed/Enhanced

Our team further developed our computer programming (backend, frontend and web application-based) skills; cyber security knowledge, methodology, and capture-the-flag related skills; setup/installation skills regarding the Linux command line and cloud-based servers; and teamwork, communication, and leadership skills throughout our team project.

3.1.3 Plans for Spring Semester

We initially planned to have a comprehensive start on the development side of the project by Spring Semester—that of programming and web-application based work. We anticipated mostly focusing on debugging the application, small design improvements, and preparations for the IT Expo for the duration of Spring Semester.

3.2 Spring Semester Conclusion

For Spring Semester, our team primarily focused on the development and testing phases of our project, using the preparations we had made during Fall Semester as a guide to work towards the presentation of our project at IT Expo.

3.2.1 Lessons Learned

The primary lesson our team learned during Spring Semester was that reaching out and asking for help from our professors and advisors was critical to our overall success. We learned that seeking advice, critique, and criticism from others was important to ensure that our assignments were completed fully, and that our work satisfied all necessary

requirements. Seeking outside advice from other teams was helpful as well, as we could then apply that advice to improving our work for Spring Semester.

3.2.2 Abilities/Skills Developed/Enhanced

Our team further developed our frontend and backend web application skills, especially that of server-side and client-side programming. On the server-side, our team gained a greater understanding of php programming and utilizing the Yii Framework to interact with the application's MySQL database. On the client-side, our team learned to work with a programming language we had never worked with before—Typescript—along with web languages including Javascript, HTML, CSS, and frontend frameworks such as Node JS and Angular. Additionally, our team learned to work with VMware virtual machines in the form of Ubuntu servers, and thus further honed our Linux command-line skills. Finally, in regards to soft skills, we continued to build upon our teamwork, communication, and leadership skills that we initially developed during Fall Semester.

3.2.3 What We Completed Since Fall

Our Senior Design project was overall split into three separate phases: Project Planning, Development, and Testing. We completed our Project Planning in Fall Semester, and Development and Testing phases in Spring Semester.

For Fall Semester, we focused on the Project Planning phase of our project—that of creating planning documentation, charts, diagrams, and scheduling, along with planning the deliverables we anticipated to complete for the project. This gave us a solid

foundation for the start of our Development Phase in the Spring. For Spring Semester, we completed the entire Development Phase of our project—that of implementing all planned functionality, coding, and general web application updates and quality-of-life improvements. Finally, we completed the Testing Phase nearing the end of Spring Semester, where we tested our application thoroughly to ensure all required implementation and functionality worked as intended.

3.2.4 What We Learned From Our Final Presentation

Due to, as of this writing, current events surrounding the COVID-19 virus, all 2019-2020 Senior Design teams were required to deliver their final presentations virtually instead of the initially planned in-person. We used a webcam video software called Zoom for this purpose. Our team held several meetings together to practice beforehand to ensure our preparedness for our final presentation. As a result, we felt we succeeded in achieving the outcomes required for the final presentation. We were given excellent advice regarding the presentation of our project at IT Expo, such as showcasing our specific changes, additions, and newly implemented functionality, alongside promoting the Ohio Cyber Range Institute's own CTF event at the Expo. We took all this advice seriously, ensuring our project's overall success at IT Expo.

4. REFERENCES

Morgan, S. (2019). Cybersecurity Jobs Report 2018-2021. Retrieved from https://cybersecurityventures.com/jobs/

Attack/Defense for Beginners. (n.d.). Retrieved from https://2019.faustctf.net/information/attackdefense-for-beginners/

Vigna, G. (2019, January 10). Capture That Flag! How participating in hacking competitions can improve the security culture of a company and the talent of the security team. Retrieved from https://www.lastline.com/blog/capture-that-flag

Brown, E. (2018, November 28). CTF Hacking: What is Capture The Flag for a newbie? Retrieved from https://www.alienvault.com/blogs/security-essentials/capture-the-flag-ctf-what-is-it-for-a-newbie

Harmon, T. (2016, September 14). Cyber Security Capture the Flag (CTF): What Is It? Retrieved from https://blogs.cisco.com/perspectives/cyber-security-capture-the-flag-ctf-what-is-it

What is Capture the Flag? (2017). Retrieved from https://ctfd.io/whats-a-ctf/

APPENDIX A. ADDITIONAL INFORMATION

1. List of Acronyms

CTF: Capture-The-Flag

OCR: Ohio Cyber Range Institute

IDE: Integrated Development Environment

VM: Virtual Machine

MVC: Model-View-Controller

API: Application Programming Interface

DEV: Development Testing

QAT: Quality Assurance Testing

UAT: User Acceptance Testing

2. Source Code

UC CTF project Github repository: https://github.com/devaughnpenn/UCCTF

APPENDIX B. TECH EXPO POSTER