

Cyber Warfare Training Platform User Manual

Team Members: John Babiak, Antawn Weg, Sarah Yo

Customer: Dane Brown, USNA Cyber Department

Table of Contents

Introduction	3
System Set Up	4
High-Level View	7
Quick-Start Tutorial	8
Detailed View of System	13
Functional Requirements Trace Table	15
Index	18
Appendices	19
a. Quick Reference Card	19
b. System Requirements	19
c. Installation	19
d. Troubleshooting/Known Bugs	19
e. Maintenance Procedures and Issues	19
f. Developers' Information	20

Introduction

This user manual serves as a reference for how to use the Cyber Warfare Training Platform (CWTP). The CWTP is designed to be used by individuals who have never done anything cyber related, and only requires that you know how to use a phone. As such, the CWTP has been designed to take care of installation, maintenance and launching on its own with minimal user input. Throughout this document, you will find steps on how to install the CWTP, how it has been designed, how to get started with it, and more.

The CWTP has been designed to be a system that can be used to train students in aspects of computer security. A student logging into this system will be brought through a series of lessons and challenges to gradually introduce them to increasingly harder concepts. The target audience is for training computer neophytes who wish to eventually join the Cyber Warfare team. Through self-paced interactive lessons, the system should bring them up to a point where they can be productive members of the team.


The CWTP is simple. It relies on the iPhone app iSH to be able to access a Linux command line terminal. This terminal is able to support many packages to include Python, SQLite, Git, and PHP. The CWTP uses iSH to host a local PHP server that is accessed through the iPhone browser. This server is updated through Github, and the CWTP updates itself every time it is run by a user. The combination of iSH and Github allows the CWTP to be powerful and easily maintained. Once the CWTP is running, it acts as a web page that the user can interact with through their mobile browser, which is intuitive for most people.

System Set Up

This is pulled from the Github page for the CWTP which outlines in a README file how to set up the system:

ic470-cwtp-pub/README.md at dev · johnbabiak/ic470...

<https://github.com/johnbabiak/ic470-cwtp-pub/blob/de...>



[johnbabiak](#) / [ic470-cwtp-pub](#) Public

[Code](#)
[Issues](#)
[Pull requests](#)
[Actions](#)
[Projects](#)
[Security](#)
[Insights](#)


dev

...

[ic470-cwtp-pub](#) / [README.md](#)


John Babiak updating README photos

History


 1 contributor

55 lines (41 sloc)

1.35 KB

...

IC470 CWTP

Cyber Warfare Training Platform

USNA Capstone Project for AY'23

Customer: Dr. Dane Brown

Team Members: John Babiak, Antawn Weg, Sarah Yo

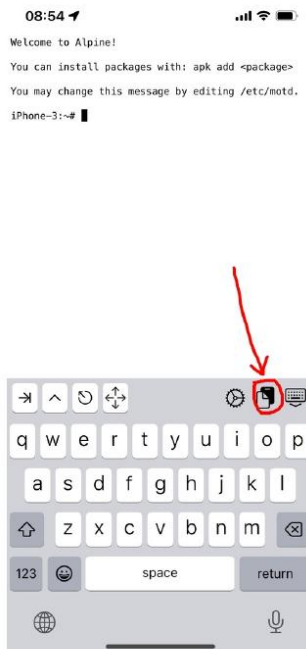
To install, copy the following line into iSH:

```
apk add git; git config --global pack.threads "1"; git clone
https://github.com/johnbabiak/ic470-cwtp-pub.git; cd ic470-cwtp-pub; git
pull origin dev; chmod +x install.sh; ./install.sh
```

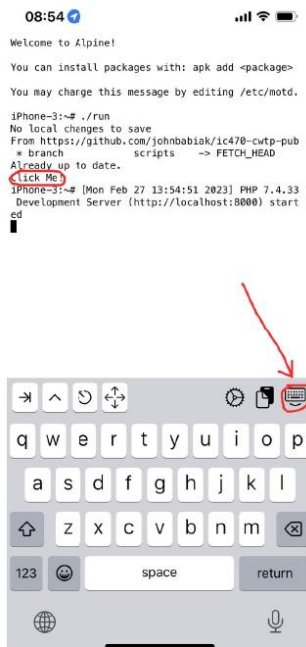
In the iSH app, you can paste using this button:

ic470-cwtp-pub/README.md at dev · johnbabiak/ic470...

<https://github.com/johnbabiak/ic470-cwtp-pub/blob/de...>



Once the script is done installing, click the "Click Me!" link to open the CWTP. You will have to lower the keyboard in order to open the link.



If the iSH app prompts you to use location services, allow it to use your location. This is so the app can run in the background and makes the CWTP work.

If you close the iSH app and want to run the system again, simply open the app and type `./run` to run the system then click the "Click Me!" link to access it.

Filenames and locations:

```
/root
- create_table.txt
- cwtp.db (created upon installation of system)
- index.php
- install.sh
- keybdwn.png
- login.php
- logout.php
- paste.png
- README.md (this file!)
- register.php
- run
- tools.php
- utils.php
/challenges
  /'Example 1'/
    - 'Example 1.php'
    - fs_example.png
/tools
  /base64
    - base64.php
  /shell
    - shell.php
```

High-Level View

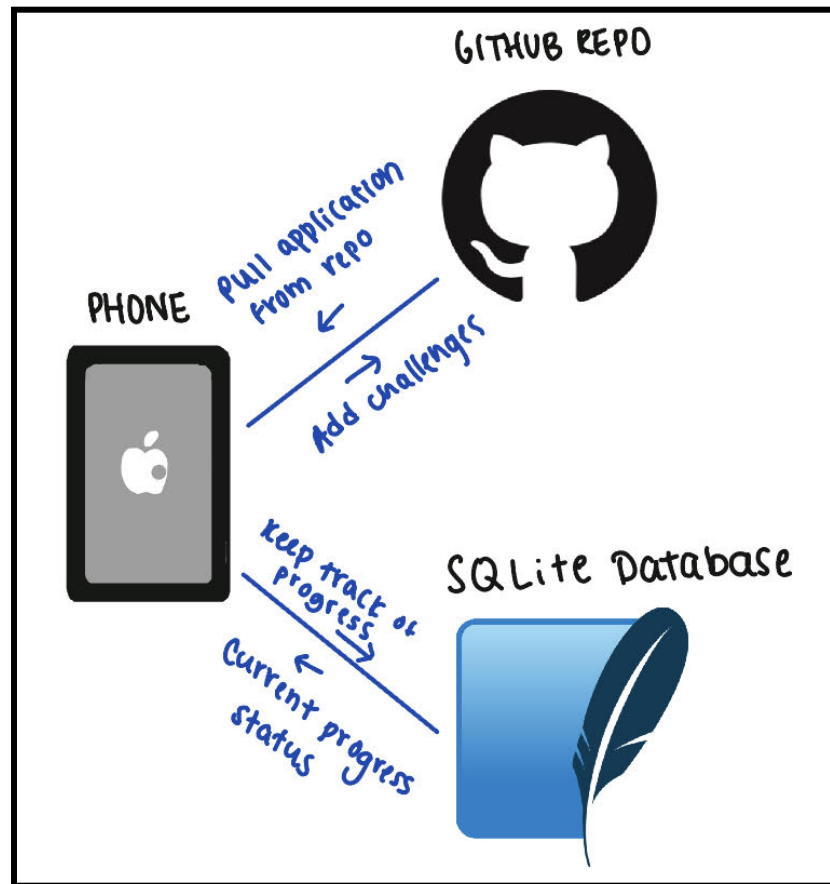


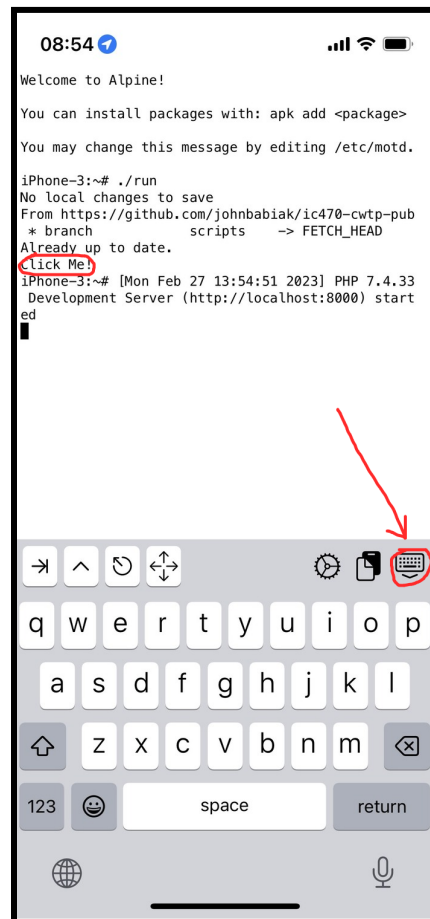
Figure 1 - High-Level System View

Looking at Figure 1, the CWTP functions on three different subsystems. The first is a Github repository which hosts the code base for the system. The next part is a phone which hosts a local instance of the code base and serves the system for the user to interact with. The third part is a SQLite database which holds user-specific information on the local device.

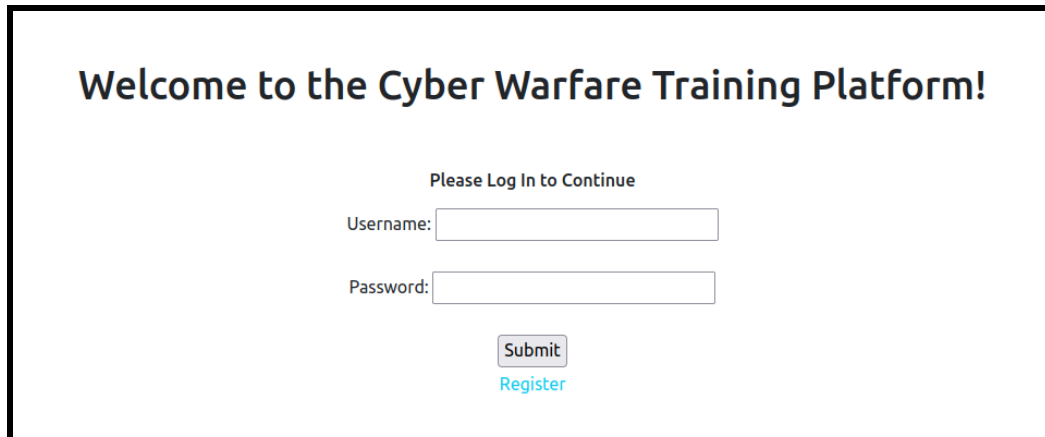
The Github repository provides the ability for the system to be updated and maintained from a central location. All local instances are able to update themselves by simply pulling from the central repository. The SQLite database is necessary in order to track user-specific details that could not be tracked by a central repository.

Quick-Start Tutorial

In order to run the CWTP, your phone must meet the following requirements. First, you must be on an iOS mobile device that has access to apps on the Apple App Store. Second, you must be running iSH 1.2.3 or newer. Third, your default browser on your phone must be set to Safari. Fourth, you must be able to enable location services for the iSH to always allow location tracking. This is so the application can run in the background on your phone. The application does not actually use or collect any information about your location.



First, start by running the system in the iSH command line and clicking on the “Click Me!” link. This will take you to the following landing page:



Welcome to the Cyber Warfare Training Platform!

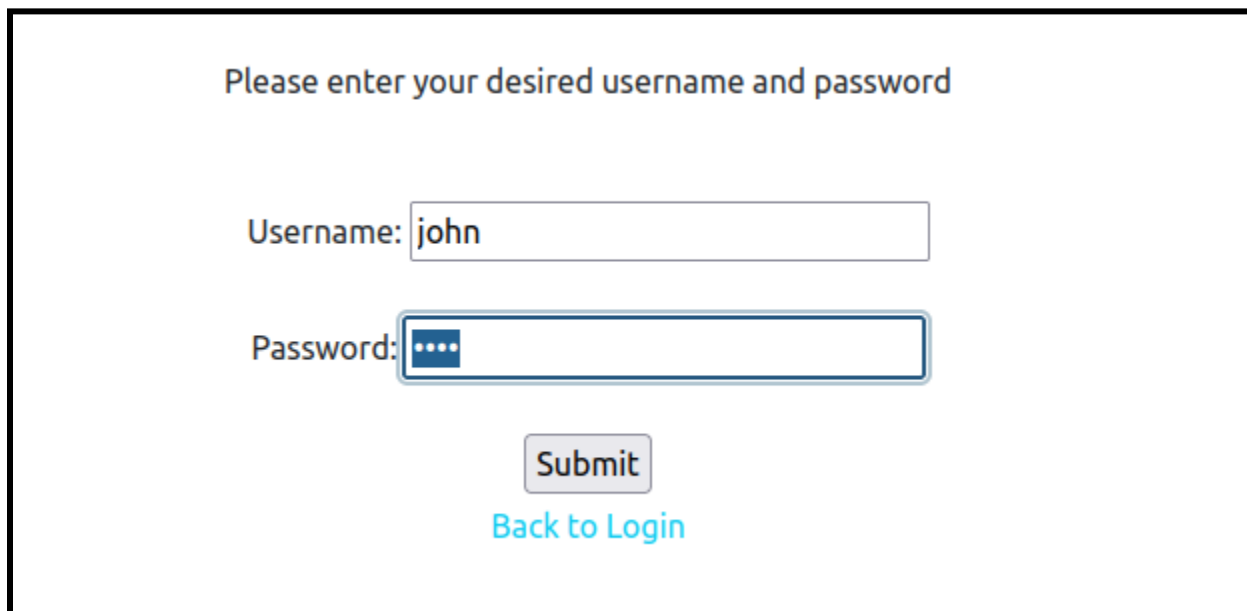
Please Log In to Continue

Username:

Password:

[Register](#)

From here, click on the “Register” link to go to the page to set up an account to log in with.



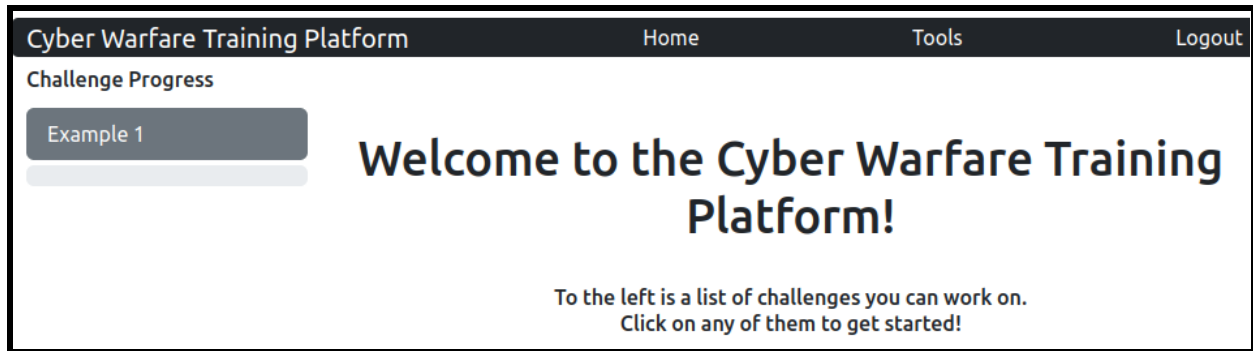
Please enter your desired username and password

Username:

Password:

[Back to Login](#)

Here you will enter your desired username and password for logging into the system with. These credentials and account will be used to track your individual progress within the system. Once you have entered your information, click “Submit” and you will be taken back to the login page. At the login page, enter your account information and then click “Submit”. This will take you to the main CWTP landing page:



Here you can see a navigation bar at the top of the page with options to take you to the home page (this page), to the tools page and to logout. On the left side of the screen is a challenge listing where you can see all of the challenges available to you and your progress on completion of the challenges. To go to the challenge, click on it in the sidebar:

Cyber Warfare Training Platform
Home
Tools
Logout

Challenge Progress

Example 1

Challenge 1

Welcome to the Cyber Warfare Training Platform!

In the world of cyber security, the BASH terminal is a common tool of choice. A terminal is a method of input to a computer that only involves entering in text-based commands, and there are no graphics present for a user to click on or interact with. It is one of the most fundamental and basic ways to interact with a computer. The BASH terminal is a specific version of a terminal with defined commands that a user can select from.

In the BASH terminal, there are two fundamental commands that are needed in order to traverse the file system of the computer. The file system is the structure of how files are laid out in an operating system. Typically, BASH terminals are associated with Linux operating systems and so we will teach you about the Linux file system.

The Linux file system is made up of two components: files and directories. Directories hold files and files hold data. The Linux file system is also hierarchical, meaning it starts at one top level directory and then all other directories are underneath that one. The top level directory in Linux is called the 'root' directory and is named '/'. Below is a picture of what a Linux file system might look like.

In order to view what directory you are currently in, you can type the command 'pwd' into the terminal and press enter. In order to view the contents of the directory you are in, you can type the command 'ls' and press enter.

This is what an example challenge looks like. If you scroll to the bottom of the challenge, you will see a quiz:

Now it's time for a quiz!

Which of the following commands can you use to view the contents of the directory you are currently in? Assume you are using a Linux operating system.

- ☐ pwd
- ☐ dir
- ☐ ls
- ☐ cd

[Submit](#)

Need a hint?

[Click Here](#)

If you need a hint for the challenge, you can click on the “Click Here” button for a hint:

[Click Here](#)

The command you should use is a shortened version of the word 'list'

To answer the quiz, select your desired answer then click submit:

Which of the following commands can you use to view the contents of the directory you are currently in? Assume you are using a Linux operating system.

- ☐ pwd
- ☐ dir
- ☒ ls
- ☐ cd

[Submit](#)

Need a hint?

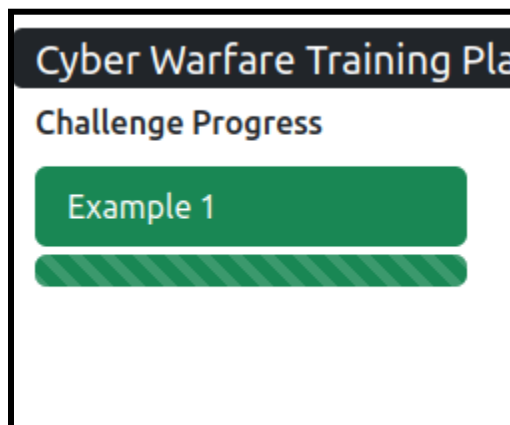
Which of the following commands can you use to view the contents of the directory you are currently in? Assume you are using a Linux operating system.

- ☐ pwd
- ☐ dir
- ☐ ls
- ☐ cd

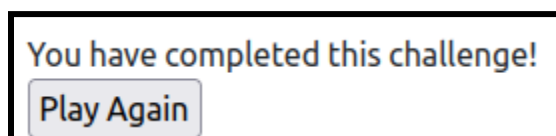
Submit

That's correct! Use 'ls' to view directory contents.

If you answer correctly, then you will get feedback that your answer was correct. Your progress in the sidebar will also update:



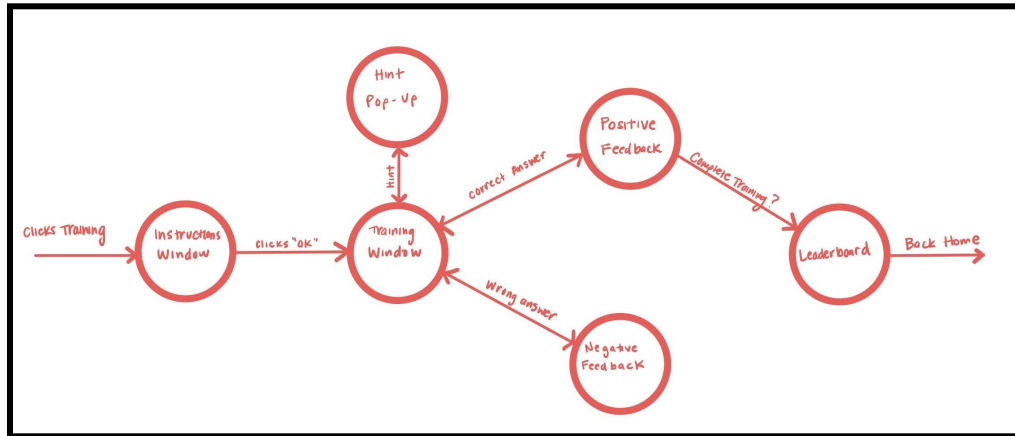
You can play the challenge again by clicking the “Play Again” button at the bottom of the page:



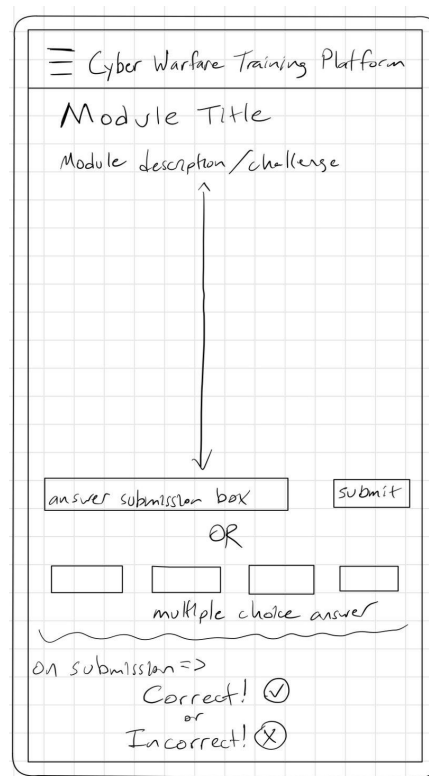
From here you can work on other challenges or log out of the system by clicking the “Logout” option in the navigation bar at the top of the page.

Detailed View of System

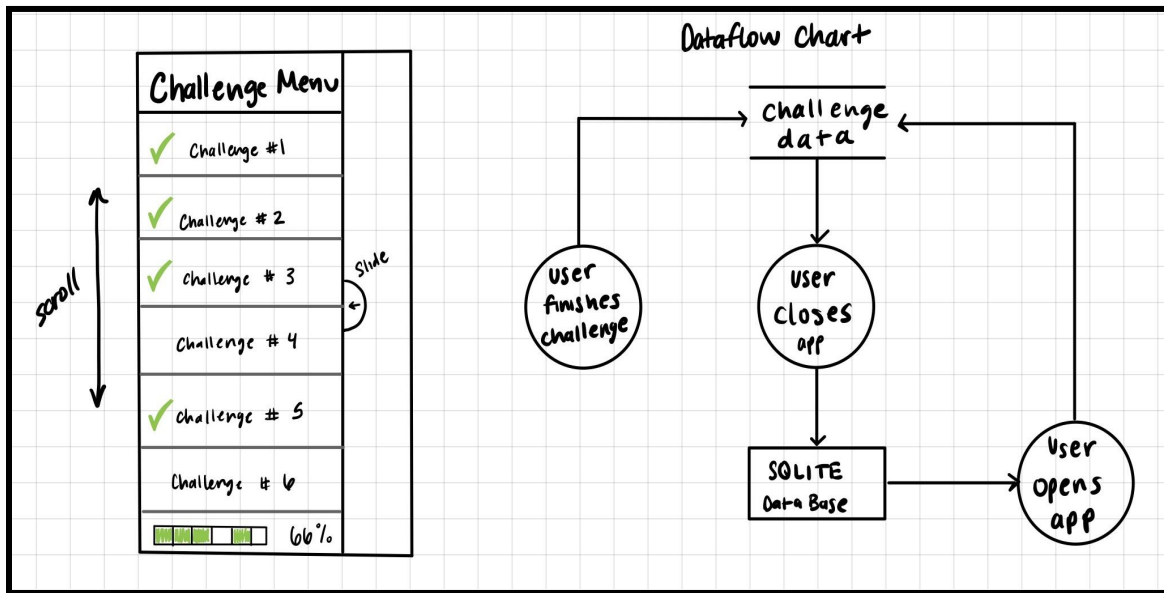
For each challenge in the system, the following state machine describes how it should operate:



So, a user who accesses a challenge should be presented with challenge instructions. In the challenge they should also be presented with the ability to view a hint. The challenge needs to have the ability to submit an answer and then get feedback on that submitted answer. Challenges should look somewhat like the following:



The system also has a sidebar that tracks progress. The following diagram outlines how this sidebar should look and also how data should flow within the system to keep track of user progress:



For more information on the system, see the high-level view of system section within this manual on page 7.

Functional Requirements Trace Table

Functional Requirements Trace Table

Functional Requirement	Acceptance Test Plan Test Cases	Build	Use Cases	Effort Value	Testing Status & Milestone #	Customer's Initials
1. Login/Password GUI: Each user shall have their own login/password pair that sets their User Role within the system. <i>Primary:</i> MIDN Babiak <i>Backup:</i> MIDN Yo	1.1 User with correct login/password is able to successfully login to the system. Expected result -> User is given the User Role associated with the login/password pair. (normal) 1.2 User attempts login with the wrong password. Expected result-> User is prevented from logging in. (exception)	Build 1		1.1: 2 1.2: 1	1.1 1.2	DB DB
2. Training GUI: Each user has a training sidebar that displays the progress of their individual cyber training. <i>Primary:</i> MIDN Weg <i>Backup:</i> MIDN Babiak	2.1 User successfully completes a training. Expected result -> An associated checkbox displays in the training sidebar indicating the training is complete. (normal) 2.2 User clicks on one of the trainings in the sidebar. Expected Result -> The user is redirected to that specific training. (normal) 2.3 Set up and learn how to use the Swift development environment to design a simple "Hello World" application. (Preliminary Step) 2.4 Learn how to design interactive elements in Swift. (Preliminary Step)	Build 1		2.1: 5 2.2: 5 2.3: 5 2.4: 5	2.1 2.2	DB DB
3. Save Progress: Each user's progress is saved before clicking out of the interface or onto a new training. <i>Primary:</i> MIDN Babiak <i>Backup:</i> MIDN Weg	3.1 User clicks away from the training or logs out. Expected Result -> Saves each individual training at its current progress point. (normal) 3.2* The system shuts down while the user is in the middle of interacting with it. Results -> The user's progress is saved within the system. (exception) 3.3 Learn how to save data on a local mobile database in Swift (Preliminary Step) 3.4 Design a database to be used locally on the mobile app (Preliminary Step)	Build 2		3.1: 3 3.2: 3 3.3: 3 3.4: 3	3.1 3.2 3.4	DB DB DB

<p>4. Create/Delete Challenge: An administrator user shall be able to create challenges for the system or, conversely, remove challenges from the system.</p> <p><i>Primary:</i> MIDN Yo <i>Backup:</i> MIDN Weg</p>	<p>4.1 Administrator uploads challenge to system to be added. Expected Result -> All users of the system can now see the created challenge. (normal)</p>	Build 3		4.1: 5	4.1	DB
	<p>4.2 Non-administrator user uploads challenge to system to be added. Expected Result -> No challenge is added to the system. (exception)</p>			4.2: 3	4.2	DB
	<p>4.3 Administrator selects a challenge and chooses to delete it. Expected Result -> The selected challenge is removed from the system. (normal)</p>			4.3: 2	4.3	DB
	<p>4.4 Non-administrator selects a challenge and chooses to delete it. Expected Result -> the challenge remains in the system. (exception)</p>			4.4: 2	4.4	DB
	<p>4.5 Design a database to store all of the global challenge information in (Preliminary Step)</p>			4.5: 5	4.5	DB
<p>5. Synchronize Local and Central Challenge Database: the system shall synchronize local apps with the Central Training Module Server challenges.</p> <p><i>Primary:</i> MIDN Babiak <i>Backup:</i> MIDN Weg</p>	<p>5.1 Administrator adds a new challenge to the system while a local app is off the Internet. The local app then connects to the internet. Expected Result -> The local app will add the new challenge to its system. (normal)</p>	Build 4		5.1: 5	5.1	DB
	<p>5.2 Administrator removes a challenge from the system while the local app is off the Internet. The local app then connects to the internet. Expected Result -> The local app will remove the challenge from its system. (normal)</p>			5.2: 5	5.2	DB
	<p>5.3 Administrator adds a new challenge to the system while a local app is off the Internet. The local app then connects to the Internet temporarily without enough time to download the new challenge. Expected Result -> The local app does not display the new challenge. (exception)</p>			5.3: 3	5.3	DB
	<p>5.4 Learn how to host the Central Training Module Server securely for mobile instances to communicate with. (Preliminary Step)</p>			5.4: 8	5.4	DB

6. Non-Internet Use: The system shall work regardless of whether or not the user is connected to the Internet. <i>Primary:</i> MIDN Yo <i>Backup:</i> MIDN Babiak	6.1 A user logs in to the system and selects a challenge to access without access to the internet. Expected Result -> The local app loads the selected challenge for the user to interact with. (normal) 6.2 A user loses internet connection in the middle of interacting with a challenge. Expected Result -> The challenge remains available for the user to interact with.	Build 2		6.1: 8 6.2: 5	6.1 6.2	DB DB
7. Sample Challenge: A sample challenge to demo to the customer to get feedback on. (Preliminary Step) <i>Primary:</i> MIDN Weg <i>Secondary:</i> MIDN Babiak	7.1 User accesses the given sample challenge. Expected Result -> The sample challenge provides instructions for the user to follow. (normal) 7.2 User enters the correct answer to the challenge. Expected Result -> The user is given feedback that they successfully completed the challenge. (normal) 7.3 User enters the wrong answer to the challenge. Expected Result -> The user is given feedback that they gave the wrong answer and should try again. (normal) 7.4 The user requests a hint from the challenge. Expected result -> the challenge reveals more helpful information to the user. (normal)	Build 1		7.1: 3 7.2: 2 7.3: 2 7.4: 2	7.1 7.2 7.3 7.4	DB DB DB DB
8. (NEW) Scripts: Scripts to install and run the challenge to make the system more user-friendly. <i>Primary:</i> MIDN Babiak <i>Secondary:</i> MIDN Weg	8.1 User enters command line single line command to install system. Expected Result -> The system is installed and runs upon completion. (normal) 8.2 User clicks link to access running instance of system. Expected Result -> The system is launched in the local browser. (normal) 8.3 User runs the system using the provided run script. Expected Result -> The system runs and provides a clickable link. (normal) 8.4 User runs the system using the provided run script. Expected Result	Build 3		8.1: 2 8.2: 1 8.3: 2 8.4: 2	8.1 8.2 8.3 8.4	DB DB DB DB
	-> The system updates itself with the current version of files. (normal)					

Index

For installing, running and getting started with the system, see pages 4 and 8 of this manual.

For design reference, see pages 7 and 13.

Appendices

a. Quick Reference Card

App to run the system	iSH Version 1.2.3 on Apple App Store
Command to install system	<code>apk add git; git config --global pack.threads "1"; git clone https://github.com/johnbabiak/ic470-cwtp-pub.git; cd ic470-cwtp-pub; git pull origin master; chmod +x install.sh; ./install.sh</code>
Github Repository	ic470-cwtp-pub
Command to run system	<code>./run</code>
How to login to system	Register account then enter in account credentials
How to access challenge	Login to system then click challenge in sidebar

b. System Requirements

The system requires that you are using the latest version of iSH on an iOS device. The iOS device must have its default browser set to Safari and location services must be able to be enabled for the iSH application. This is not to collect or track information about the device's location, but rather as a loophole to run the application in the background of the phone.

c. Installation

See page 4 for system set up

d. Troubleshooting/Known Bugs

For troubleshooting, first try to reinstall the system by deleting iSH, reinstalling it and then going through the installation steps. This will delete all user progress in the system. For other problems, submit an issue to the Github repository for the system.

e. Maintenance Procedures and Issues

The system maintains itself by pulling from the Github repository every time the system is run. If the system seems to be out of date, try to restart it by closing iSH, reopening iSH and then running the system. To edit the code base within the repository, please submit requests to the ic470-cwtp-pub Github repository.

f. Developers' Information

John Babiak
johnbabiak531@gmail.com
512-412-1350

Antawn Weg
antawn1998@gmail.com
605-759-6911

Sarah Yo
skyo4046@gmail.com
813-997-7660