

## **Bermuda: Gateway to Cybersecurity Challenge Platform**

Project Plan

Python Pirates

Authors: Emily Clauson, Jake Khal, Stephen Swanson, Alexandr Iapara



<https://github.com/Fenix-A-I/Project-Bermuda>

Professor Juan Flores

CS 422, Fall 2024

## Application Overview:

Capture the flag events are commonly used in the cybersecurity world to learn about real life vulnerabilities. They provide users the chance to practice ethical hacking skills in controlled environments. “Hack the Box” and “VulnHub” are common platforms that are used for this, but usually require money in order to participate. The University of Oregon Cybersecurity Club needs an educational platform such as this that is free and safe for students to use. Currently, no system within the university offers this, creating a gap in cybersecurity skill development. Bermuda is a gateway that will act as a foundation for a capture the flag experience. It will securely authenticate users, and store credentials for them to have their own linux environment, and track progress in challenges. A draft of what the user interface would look like is presented below.

The image shows a two-part draft user interface for the UO Cybersecurity Club. The top section is a dark gray rectangle containing a centered white box with the text "Welcome to UO Cybersecurity Club". Below this is a blue button with the text "Click here to sign in (use UOregon email)". At the bottom of the white box is a small Microsoft 365 logo. In the bottom right corner of the top section, there is small text that reads "Made by members of Project Bermuda" next to a small icon. The bottom section is a larger dark gray rectangle with the text "UO Cybersecurity Club" at the top. Below this is a large white rectangular input field with the placeholder text "Paste your SSH key here...". At the bottom center of this section is a blue button with the text "Submit SSH Key".

## Management Plan

The python pirates have split the project based off of modules required. These modules consist of both the front (client side) end and the back (server side) end. The team primarily communicates via discord, and tasks are assigned via scrum. Each week the team has a meeting breaking down the success and challenges of the week before, what tasks they can handle for the week, and any help they might need. In these weekly meetings we also assign our tasks for the scrum sprint. We assign tasks based off of sprint points where one point is equal to one hour of work. All work is committed and pushed into github with exceptions of the project plan and presentations which are written in google drive. The team also has daily stand up posts where each member reports progress of the day before, current day, and next day. Below is a general layout of modules and documents with which team member took responsibility for them. More information on these modules can be found in the SDS.

#### Project Proposal (all)

- SRS
- SDS
- Project Plan

#### Communication with Professor and Turn Ins - Emily

#### Authentication - Emily

- Microsoft OIDC implementation
- Secure authentication of only University of Oregon Accounts
- Documentation

#### Web Interface - Jake

- HTML pages for:
  - Landing/Welcome page
  - Link to authentication
  - Configuration page
  - Challenges pages
  - Terminal pages

#### Database - Stephen

- Secure storage for credentials
- documentation

#### Flask Implementation - Alex

- Source for connecting HTML to back end
- Deployable pages for each HTML source

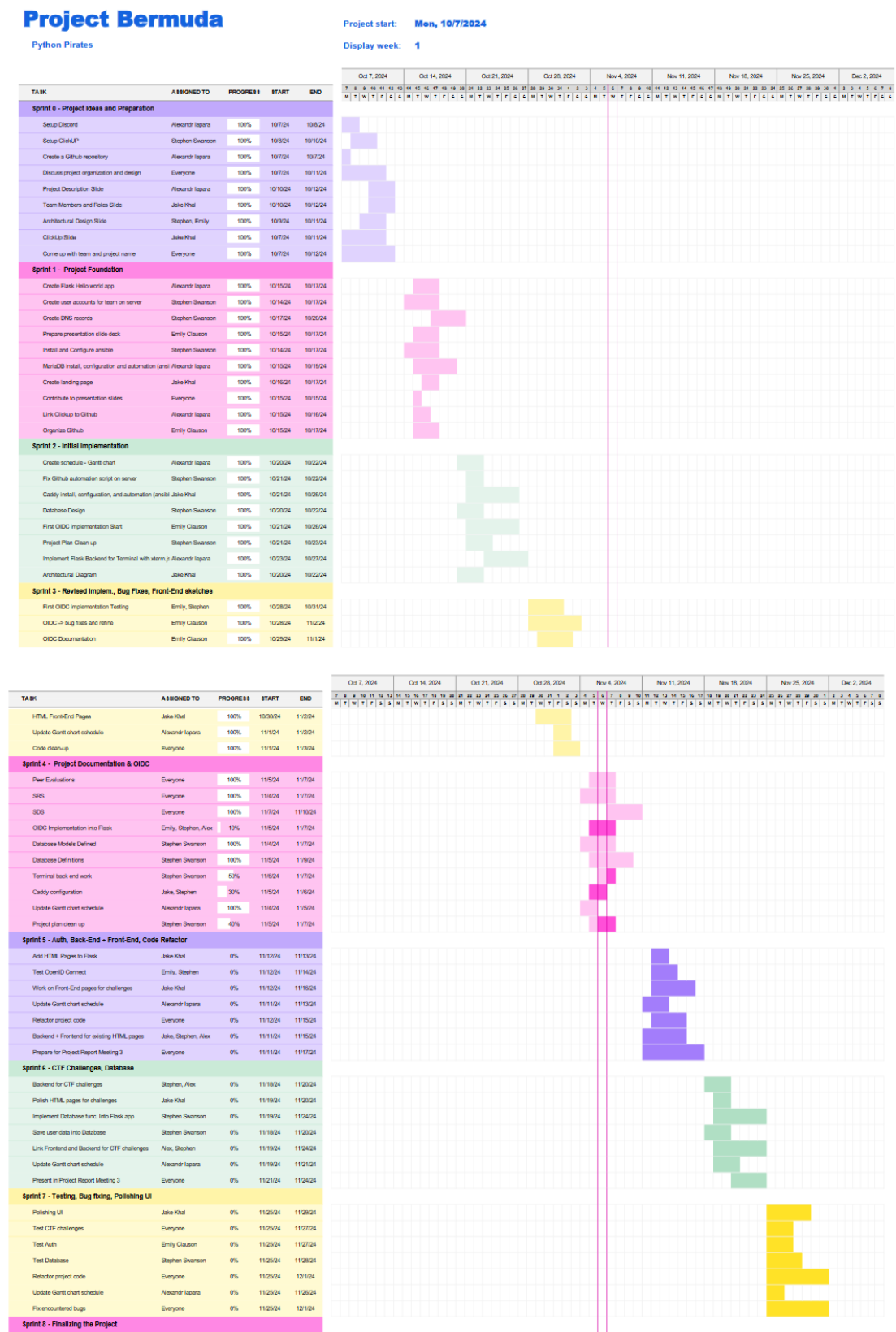
- Documentation

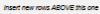
## Work Breakdown Schedule


Below is a general breakdown of the project:

Project schedule	Work breakdown schedule
October 9th	Team meeting 1
Oct 11th	Proposal Rough Draft
Oct 15th-17th	Project Plan presentations
Oct 14th - October 20th	Sprint 1
October 23rd	Progress meeting 1
Oct 21st-27th	Sprint 2
Oct 28th - Nov 3rd	Sprint 3
Nov 6th	Progress meeting 2
Nov 4th - Nov 10th	Sprint 4
Nov 11th - Nov 17th	Sprint 5
Nov 19th-21st	Progress meeting 3
Nov 18th - Nov 24th	Sprint 6
Nov 25th - Dec 1st	Sprint 7
Dec 3rd-5th	Project deliveries
Dec 6th	Project Due

Below is a Gantt chart that describes more details of the schedule above:







**Alex** 10/18/2024 1:12 PM

Yesterday:


- Added some front end pages and fixed flask issues
- Learned and played around with ansible, configuring it to install MariaDB
- Gained access to the server

Today:

- Make flask run on the server

Help needed:

not atm



**Jake** 10/18/2024 2:27 PM

Yesterday:

- completed task

Today:

- researched xterm.js

Help needed:

none

## **Building Plan**

The project “builds” will be implemented in coordination with our sprints on scrum. The first sprint consists of research in order to prepare for initial implementations, as well as start our proposal consisting of the project plan, SRS, and SDS. Sprint two consists of initial implementations of each person's assigned component. These are all done on separate branches to ensure there are no conflicts. Sprint three and four consist of debugging, writing documentation, and tuning initial implementations. These stages are highly collaborative with the group so that each member can receive feedback and help for their assigned module. Sprints five and six consist of combining all individual components. These are the sprints in which we plan to have the first functional implementation of the project. We will merge branches of the project and edit code to allow a working environment. Sprint 7 consists of fine tuning and debugging our code to meet the requirements set. This last sprint will also allow us time to polish documentation for the project. Through these stages we will have progress report meetings with Professor Flores and Hummad to check in and discuss possibilities to flourish. The dates of these sprints are found in the work breakdown schedule above.

## **Rationale**

The architecture is modular to enhance scalability and security. By isolating each module such as authentication, database management, and challenge handling, each component can be independently optimized or updated. There are no components that are fully dependent on another, meaning the individual pieces can be brought together after they are worked on independently. This also ensures a clear separation of concerns, minimizing the risk of system failures or security breaches. Our management allows each team member to clearly know their expectations as well as reach out to other team members for assistance. It allows a collaborative setting with concurring tasks happening at once for an efficient response time.