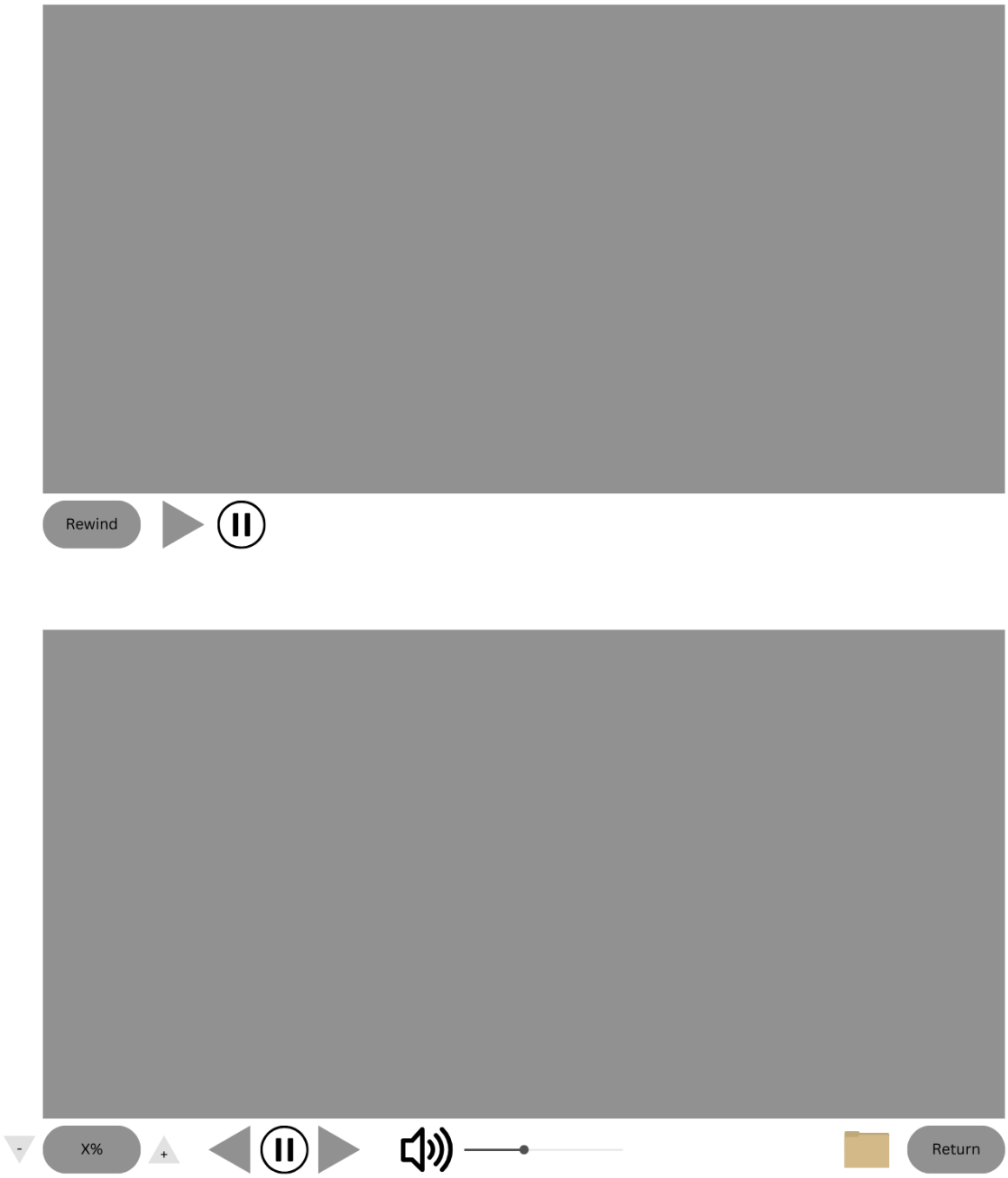
The Goobernauts
Names: Stephen Goodridge, Logan Geiser, David Geng, Ryan Giles, Heath Miller
Date: 10/15/21

UI Prototype:



Problem Statement

Our project plans to provide an open source replay system for referees in the USA Fencing Team which is capable of automatically and manually recording, replaying, and saving the video for later analysis. The current recording system Erplay used by the USA Fencing Team is a closed source software, which only operates on Mac devices limiting accessibility and improvements while costing fees for its usage. This, in practice, means that the referees and judges are required to use a Mac device that has not been updated past a certain system update, the outdated software has not been updated to support modern OS versions of mac and limits the possible devices that can be used. The system shall be able to autonomously capture video when signaled to, replay the footage at selected speeds, and capture the video at no less than 60 frames per second. It should also be able to run on at least current versions of Windows and Mac

Design Explanation

For our initial concept, we thought of a basic video player with options to manually initiate recording, rewinding, and play/pause features. The video feed should be the most prominent in the UI, and any other features, while visible and easily accessible, should not be distracting or get in the way of the video feed. We chose this design as the project focus is mainly on the functionality of the program with minimal requirements in UI design and functionality besides the initial video player and rewinding.

Prototype Implementation

The next steps we plan to take is implementing and testing webcam functionality and basic functionality and support with Python or C/C++ libraries. We are unsure how Python, as an interpreted language, will handle camera output, especially when frame rate and quality are so important to this project. Prototyping in Python will help us learn its limitations in the context of the replay system, as well as help us better understand methods for designing and implementing features if we do turn to C/C++ instead.

Implementation Plans

1. Get input from laptop webcam and display feed on screen.
    a. Python Library
2. Implement a basic UI to test function readability.
    a. Python Library
3. Script keyboard inputs so app operation can be done quickly
4. [Get Feedback from Terry]

Potential Improvements

Possible improvements could be focused on usability, organization, and design of the user interface. For example, a friendlier UI to ensure judges who are less technologically inclined are able to operate the system. Better button organization or cutting down on the clutter they cause could improve the appeal of the UI as well. The design itself should ensure any judge is able to understand, operate, and navigate the interface which leaves room for possible improvements for the UI, but overall will likely remain rather minimal as the user interface is mainly to interface with the video feed.

Client Feedback

1. Is the UI design an improvement on the old system?
2. What other additions would you like to have on the UI?
3. Is there anything you'd like removed from the UI?
4. Is the UI design intuitive and easy to use?
5. Any suggestions on potential improvements on the UI?