CyberConnect Protocol and

Content Platforming

## Jonathan Yakimow, Timothy Connor, Aidan Schneider, Simeon Bauman

## [-Based on the official CyberConnect Dev Center Guide-](https://docs.cyberconnect.me/how-to/build-content-app/quick-setup)

We closely followed the official guide (linked above), and there were admittedly places where we felt it was either lacking or unclear. The CyberConnect content platform runs on the React: Node.js framework, and we successfully deployed a server using the test code they provided through their [GitHub](https://github.com/cyberconnecthq/). We struggled for a while attempting to figure out how to set the environment variables, but later discovered that they were already configured automatically.

We spent some time editing the graphical interface as well, changing colors and fonts of backgrounds and interactable elements. This proved to be quite easy – essentially everything was at our fingertips from the globals.css file located within the styles folder. We were also able to create a few new pages on the website after some trial and error: one page successfully pointed to a YouTube link, and the other was supposed to display an image, but we were unable to accomplish this.

We had some troubles in minting profiles for multi-user integration, but after several befuddled hours, we realized that the name of the file that held our environment variables was simply supposed to be called “.env”. This was not made clear in the slightest by the guide we followed. All things considered, the environment variables were a huge stumbling block during our entire creative process.

This project was a great learning experience for all four of us, as none of us had done very much work with distributed ledger technology, MetaMask, or the Node.js framework. We even were able to give the website Concordia-themed colors! We are pretty proud of what we were able to accomplish within the limited time frame we were allotted.