Videos used: https://www.youtube.com/watch?v=SNyQu_FXkqs
https://www.youtube.com/watch?v=ORb63TtbflQ
https://www.youtube.com/watch?v=TwYRh0E4cII
https://www.youtube.com/watch?v=yufqeJLP1rI
https://www.youtube.com/watch?v=kPTJ1uYLjgE
https://www.youtube.com/watch?v=-TUEM2NCuVE
https://www.youtube.com/watch?v=SCS--9dZ6w8

Notes

Since social media limits data available to scrape if one is not logged in there are two ways to still get this data.

Using a proxy server will trick the website into allowing scraping still, but is something that websites don't like and there is a question about its legality.

One can also log in and use their own account, but since these websites don't like web scraping there is a risk of an IP ban and/or even issues with one's account if we use this option.

Utilizing a proxy server would mean that the code would be very similar to our other web scraping code, just with code to connect to a proxy server.

Residential rotating proxy server is the best for web scraping if we were to use a proxy server.

Proxy usage also has the downside of a possible cost.

In terms of connecting an account to social media to scrape data, we may want to use auth0, which has built in sign-in functionality, including to third-parties like google, facebook, twitter, etc

Auth0 seems to have built in customizability in allowing users to have differing levels of login requirements (eg. password, 2fa, fingerprint, etc)

Auth0 is mentioned in relation to NextJS as opposed to NodeJS, which may mean some minor differences between setup for our project if we were to use it

If we were to try and connect accounts using python, django looks like it is best, but it seems that django isn't really something that we can utilize in our project, due to our project using different systems

If we were to use django, it seems like we would need to utilize a cloud service of some sort, which may not be in the scope of the project (though there has been talk of doing this in the past, so it is not impossible such a thing would be desired)

The best choice would likely be to implement Auth0 (or an equivalent app) in NodeJS and have people sign into their account in order to select data to scrape.

To implement Auth0 in Nodejs, you:

1) Install dependencies
   ————————————

   npm install express express-openid-connect –save

2) Configure router (to be placed in code)
   ————————————

```
const { auth } = require('express-openid-connect');

const config = {
  authRequired: false,
  auth0Logout: true,
  secret: 'random secret' (can generate using openssl rand -hex32 in cmdline)
  baseURL: 'https://localhost:8080' (change as needed)
  clientID: 'clientIDhere' (change as needed)
  issuerBaseURL: 'https://dev-gb8efxm6.us.auth0.com' (change as needed)
};

// auth router attaches /login, /logout, and /callback routes to the baseURL
app.use(auth(config));

//req.isAuthenticated is provided from the auth router
app.get('/',  (req, res) => {
  res.send(req.oidc.isAuthenticated() ? 'Logged in' : 'Logged out');
});
```

Essentially, it seems that connecting auth0 to nodejs won't be a problem and is in fact a fully supported feature. It seems like the best option for implementing our feature.