Project Report

Team: Ngineers

Author(s): Rob Ronayne, Joshua Young, Jinchun Tang, Justin Hogarth

Objective

The goal of our project is to implement HTTP authentication by using Google OAuth API as well as our own authentication where users can create their own username and password. Custom users on our platform have their password encrypted in the credentials file for security purposes. Only authenticated users can perform create, update, and delete operations of the API handler, but everyone can use get operation. Authenticated users also have a short profile associated with their information that is displayed on a new page. In addition, they have access to an api handler console which assists in performing CRUD requests. The reason we are doing this is that our server currently lacks any kind of security, and users can feel more secure by using our server. Although we assume the users are responsible, additional logging is used for tracking users' operation records.

Specific Contribution

For the most part, we built our project on our own. All the handlers, which include the profile handler, authentication handler (with three endpoints /login, /signup, /logout), api console handler, and the updated api handler, as well as their associated factories and tests, were added to the server ourselves. Additionally, adding persistence as well as adding the use of a user profile struct, a profile getter function, and a profile manager struct were all also done on our own. Finally, for the most part, the three html pages: login.html, signup.html, and console.html, were created on our own as well.

The only things that were pulled from additional sources were the use of Google OAuth and how to incorporate it into our codebase (found here), which is credited in the login.html page, and the use of OpenSSL to encrypt passwords (found here), which is credited in profile_manager.cc. Additionally, we modified a HTML template (found here) in order to present a cleaner interface to users. Of course, the Boost library and all other supporting libraries built into the cs130 docker image were utilized as they have been throughout this quarter.

Tools

None of us were familiar with Google's OAuth service so we looked into the proper workflow to authenticate users. This involved learning how to access the OAuth server and parse JSON Web Tokens using Google's Javascript library. We were also unfamiliar with utilizing browser cookies to maintain sessions and had to read supporting documentation on how to properly write and delete cookies to manage user login status.

To encrypt the passwords for our custom user signup feature, we had to learn about utilizing OpenSSL for encryption and decryption. Finally, to improve the responsiveness and appearance of our webpages, we had to view some examples for adding CSS and Javascript to static HTML. We also had to read about processing user inputs and creating proper HTTP requests pointed at our server endpoints.

Methodology

Authentication Handler (contains sub-endpoints for login, signup, and logout):
https://code.cs130.org/plugins/gitiles/ngineers/+/refs/heads/main/src/request_handler/authentication_handler.cc

Authentication Handler Tests:
https://code.cs130.org/plugins/gitiles/ngineers/+/refs/heads/main/tests/authentication_handler_test.cc

Profile Handler (serves the static profile page):
https://code.cs130.org/plugins/gitiles/ngineers/+/refs/heads/main/src/request_handler/profile_handler.cc

Profile Handler Tests:
https://code.cs130.org/plugins/gitiles/ngineers/+/refs/heads/main/tests/profile_handler_test.cc

API Console Handler (serves the static console page):
https://code.cs130.org/plugins/gitiles/ngineers/+/refs/heads/main/src/request_handler/api_console_handler.cc

API Console Handler Tests:
https://code.cs130.org/plugins/gitiles/ngineers/+/refs/heads/main/tests/api_console_handler_test.cc

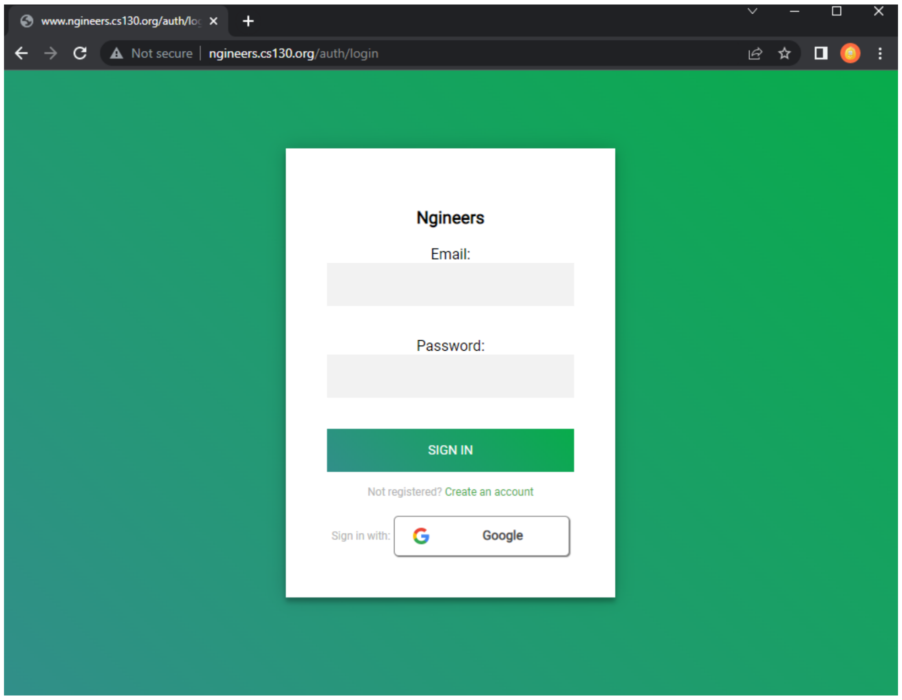
Profile Manager (manages user login, signup, and logout):
https://code.cs130.org/plugins/gitiles/ngineers/+/refs/heads/main/src/profile_manager.cc

Profile Manager Tests:
https://code.cs130.org/plugins/gitiles/ngineers/+/refs/heads/main/tests/profile_manager_test.cc

Evaluation

Below is a brief guide on how to interact with the various features implemented in our project. Note that all links provided below point to our server running in production. To test this locally, please follow our readme located here to build the binary and replace 'www.ngineers.cs130.org' with 'localhost:8080' in all the links provided below.

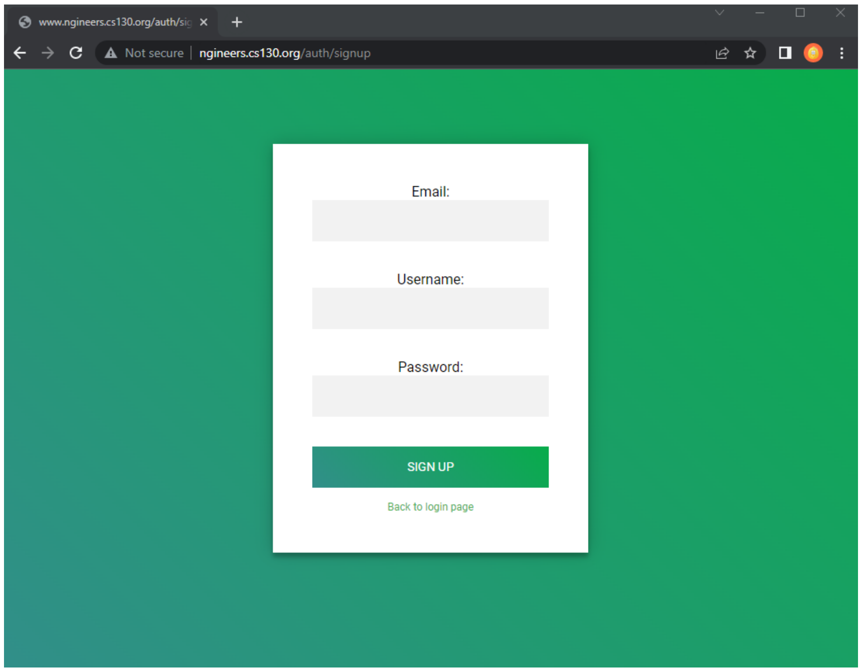
Login page: http://www.ngineers.cs130.org/auth/login



www.ngineers.cs130.org/auth/log
Not secure
ngineers.cs130.org/auth/login
Ngineers
Email:
Password:
SIGN IN
Not registered? Create an account
Sign in with:
Google

Users can sign in through Google by clicking the Google sign in button. Also, users can create an account by clicking the Create an account link and sign in with their email and password. (Important Note: Since we are accessing Google's OAuth service through a http origin instead of https, the project cannot be set to 'in production' which allows all Google accounts to authenticate. Instead, we are limited to a set of test users who must be allowed manually. The professors, graders, and TAs should have been added to this list. In the event that the OAuth login fails, please notify one of the team members so we can add you to the list.)

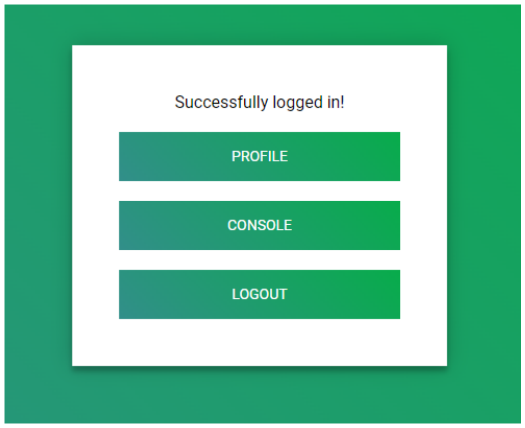
Signup page: http://www.ngineers.cs130.org/auth/signup

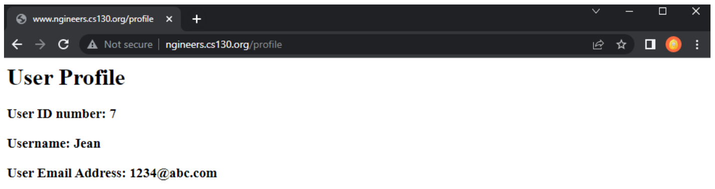

www.ngineers.cs130.org/auth/sig
Not secure | ngineers.cs130.org/auth/signup
Email:
Username:
Password:
SIGN UP
Back to login page

Users can sign up with their email address, username, and password. After filling out this information, click the sign up button, and it will prompt them to the login page.

User profile page (Only accessible after login): http://www.ngineers.cs130.org/profile

After logging in, there will be three options displayed. The logout button will sign the user out and return them to the login page. The user can click profile and the profile page will show the user's information which includes id, username, and email. Each user’s ID number will be unique to every user. Finally, the user can click the console button to view the api console page.




www.ngineers.cs130.org/profile
Not secure | ngineers.cs130.org/profile

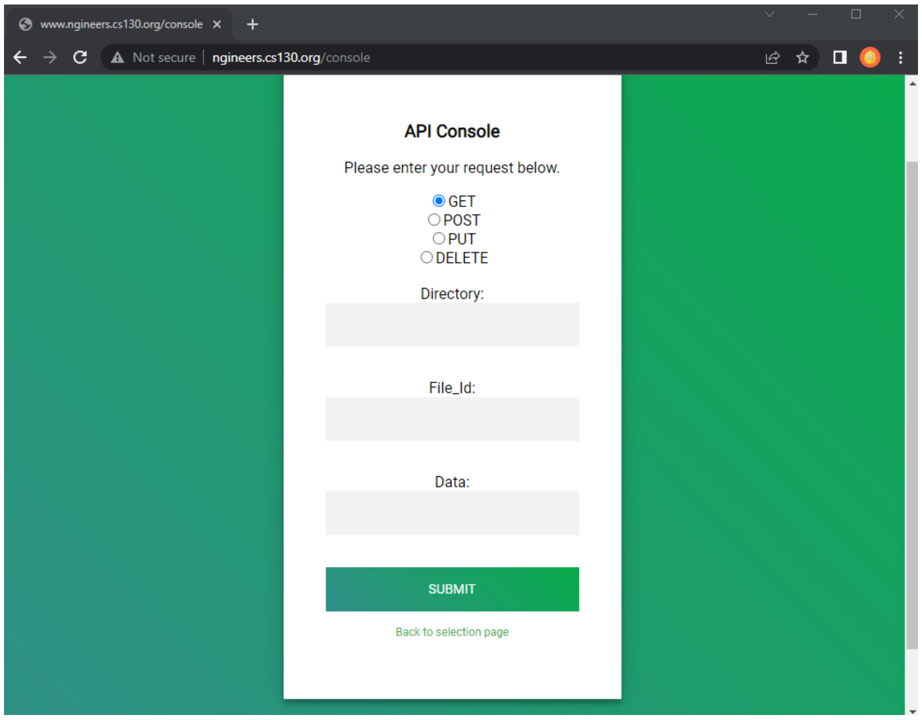
User Profile

User ID number: 7

Username: Jean

User Email Address: 1234@abc.com

API console page (Only accessible after login): http://www.ngineers.cs130.org/console


www.ngineers.cs130.org/console
Not secure | ngineers.cs130.org/console
SUBMIT
Back to selection page

API Console

Please enter your request below.

[✓ GET]
POST
PUT
DELETE

Directory:

File_Id:

Data:

Authenticated users can perform CRUD requests on this page. Select an operation and type in information for Directory, File_Id, and Data, then click submit.