**CarpoolNU Website Suggestions**Tianye

Hi Professor Kras-Morgan, to show some understanding and able to improve the reliability of the website, I first did a scan and testing on the website and came up with a few potential improvements for the current vulnerabilities. Then I also spent time initializing and running the project locally and refactored some code and attempt improve the usability, in parts like authentication session timeout, data encryption and injection prevention, input error feedback and etc…

(Here the injections response like SQL, XSS are not precise since this automatic scan didn’t prepare specific payloads, so there could be other problems beside listed here. If needed I can spend more time for more comprehensive tests, with other tools and manual injections)

**A screenshot of a computer

AI-generated content may be incorrect.Figure 1: Initial basic scan via Wapiti**

**A screenshot of a computer

AI-generated content may be incorrect.  
Figure 2: Testing result via ZAP (Zed Attack Proxy)**

**Part One: Vulnerabilities, Issue One: ClickJacking**

This first issue is basically

Here I created a as well as a header check with the get method(which returns null for the x-frames settings which indicated the vulnerability exists)

**A screen shot of a computer code

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

A screen shot of a computer program

AI-generated content may be incorrect.

Suggesting Solution: Set X-Frame-Options

X-Frame-Options: DENY / Content-Security-Policy: frame-ancestors 'none'

(Here is the result of the next day I ran the project locally and made the change)

**Issue Two: Lack HSTS Strict Transport Security – allowing SSL Stripping and other Man-In-The-Middle Attacks**

A computer screen with white text

AI-generated content may be incorrect.A computer screen shot of a computer

AI-generated content may be incorrect.

Here we could see neither the initial redirection page(307 request) nor the sign in page(200) has demonstratiied

Where in for example, using a public wifi

He or she could try intercept the http request from the target, strip the encryption (https -> http) off the victim’s request, and able to see the transmitted data(if target’s not careful enough to notice the url change)

Suggesting Solution: **Add STS setting in the response header**

“Strict-Transport-Security: max-age=31536000; includeSubDomains; preload”

Specifically in the code, add module.export, in the asych-headers return headers including -> {key: "Strict-Transport-Security", value: "max-age=31536000; includeSubDomains; preload"}

**Issue Three: CSP missing – allowing external files (such as scripts, iframes, images) and other resources to load without restriction.**

As described by the title this issue is caused by the missing of Content-Security-Policy settings. Here I swapped the button from the

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Suggested Fix: Similar to the previous issue, we can add security header, this time with Content-Security-Policy Settings.

A screen shot of a computer program

AI-generated content may be incorrect.A screen shot of a computer code

AI-generated content may be incorrect.

**Issue Four: X-Content-Type-Options missing – allowing browsers perform execution of unexpected or mislabeled files**

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a crop image

AI-generated content may be incorrect.

Here I created a fake image jpg file (containing JavaScript command), and changed extension to jpg and successfully uploaded it. However, it kept on loading at the crop image page, I realized this intermediate process somehow

(In short, this issue’s risk is mitigated by frontend logic(crop image), but neither purposely designed nor able eliminate future risks, if later add other file upload functions with no intermediate steps, it could   
  
Solution:

**Other Minor Issues:**   
Besides the aforementioned vulnerabilities, there are several minor issues regarding the current library version or the information exposure in headers.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

So it says here the current library of our website (v12.3.4) is quite vulnerable, with exploitable such as authorization bypassing and middleware sub requests. I checked and the newest version is 15.3 (released in this April), so it would be best to update the library to the newest version

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a message

AI-generated content may be incorrect.

Also, several important data , since we already open sourced this project on github, since its already being documented publicly

However if in future there’s consideration to , w hide information like this on the HTTP response headers.

**Running the project and Making Improvement Attempts**

After performing the scan and researched solutions for each vulnerability, I found the github source code <https://github.com/sandboxnu/nucarpool> (not sure this the latest version, but I think the commits are rather recent and the interface resembles the current deployment so I’ll just proceed with making modifications )

So in this section I’ll follow all the steps to deploy it, and check the source code proposed a few potential improvements, base on the source code and

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.  
My Mapbox Token: pk.eyJ1IjoidGlhbnlld2FuZzEyMyIsImEiOiJjbWNzbGRxdzMwMm15Mmpwbndyc2ljam92In0.9Ehqc9krc6xeVZD6iPE8LQ

For the AWS Tokens I just tried to bypass the logic and completed the signup session without it.  
A screenshot of a document

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

(By the way, here the Terms and Conditions disclaimer seems to be AI-generated. Not saying it’s not okay, but if needed I could try to research and rewrite it to make it more custom and precise)

**Suggestion for the Authentication:**  
For the authentications I noticed it does not have a session timeout and automatic logging out,

A screenshot of a upload profile picture

AI-generated content may be incorrect.

**Data Safety  
Suggestion one: Encryption**

I checked and noticed that currently the user data are retrieved and stored in plain text. It would be beneficial to   
here I just added the encryption.ts in the utils folder, and

A screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

A computer screen shot of text

AI-generated content may be incorrect.

Then here in the users.ts,

A computer code on a black background

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

For example if we consider the company’s address as sensitive, and modified code as shown above, we could see that the newest input is able to be encrypted and stored in a hexadecimal format.  
  
**A screenshot of a computer

AI-generated content may be incorrect.**

**Suggestion Two: Additional SQL Injection Prevention from the Back End**

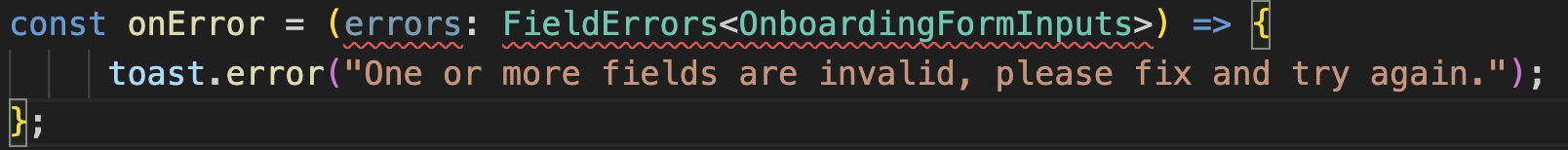
Since my last project using PHP and MySQL had potential SQL injection problem (I managed to solve it with PHP’s prepared statements), here I saw the tech stack include MySQL and decided to check it as well (like on my previous screenshots the username and other fields). None of it worked and I double checked, I think the Prisma layer is able to completely parametrize the inputs and prevent this kind of issue. So at this time it could not be a big issue, if needed for extra safety, I think we can add for example in the backend zod schema checks like:

“fieldName: z.string() .max(191, “no longer then 191 characters”) .regex(/^[^’;--]\*$/, "cannot contain SQL keywords”)” to further block all the potential SQL keywords.

**User Experience and Input Feedback**

When editing and submitting the changes on user profile, I also noticed the error message would always be “One or more fields invalid, please fix and try again.” It took me quite some time to figure out where exactly is causing the error.

I simply refacter the onError function to iterate through all the fields and append the exact invalid field in the toast output. (Could make this more precise and modular in the future.)

Before ->

A close-up of a message

AI-generated content may be incorrect.

After ->

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer error

AI-generated content may be incorrect.

Hi Ms. Kras-Morgan, so that’s the end of my analysis and suggestions I was travelling last week and just worked on this over the weekend (I have ideas on some other aspects but don’t have time to finish them yet). I’m very interested in working on this project, looking forward to hearing back and learning potential next step of this application (or other implementations for me to fulfill).🍁