# Assignment Cover Sheet

To be completed **electronically** by the student and submitted with each piece of work. Please upload this completed cover sheet via Turnitin

**Assignment Title:** Web Dev II (Multi Device Application)

**Tutor:** Usman Ahmad

**Student Name: Antonio Miguel Intal**

**Student Number: 521517**

**Date of Submission: 09/01/24**

**Details of your submission**

For an online submission, please enter the URL of where your project files can be accessed (e.g. Google Drive). If a physical artefact, please provide further details on how this project has been submitted to tutors.

*GitHub Repository Link*:

https://github.com/Antonio-III/codeguardian

Live Link of your website, GitHub/Netlify:

https://antonio-iii.github.io/codeguardian/

YouTube Video Link:

https://youtu.be/UwQhQp3KAis

In submitting this assignment, I am confirming that I have read and understood the regulations for assessment, and I am aware of the seriousness with which the University regards unfair practice. Please see the universities Unfair Practice Policy for details.

Signed: antonio Date: 09/01/24

# Development Document

## Brief

We were tasked to create a website that can be of these categories: “Promotional Site”, “Art Experience”, or a “Utility Tool”. I chose the utility tool. The implementation of this tool is to create a “password generator”.

### Specification

The password generator focuses on the responsiveness of generating new passwords. What this means is that when you click on one of the conditions to generate a password, it will generate a new one. But this is not as important of a feature as there are many password generator sites that have this as well.

A unique feature of this site is the fact that there is a visual response whenever a new password is generated. This is represented by the green thin line below the generated password, but above the options. Most sites[1][2][3] do not share this feature so this is unique to the specification of this site.

Another unique feature is that whenever the user clicks on the password, it automatically clicks the entire string. This is equivalent to double-clicking a text but it is done in only a single click. Most sites do not share this as well, so this is also unique to this website.

## Design

The layout is designed using flexbox. The flow is column. This is how the elements are arranged from top to bottom. This is because I wanted the website to be easy to look at. At a glance, you’ll be able to see everything on the site.

Navigating the site is very straightforward and intuitive, and it is responsive in a way that things get changed around the site depending on the device width of the user. For instance, notice the github icon on the top-right, if you’re accessing this on a tablet, that icon will be moved to the bottom. This is a design choice as I wanted the top-center to be the icon of the site.

Another thing you'll notice when the width has gotten smaller is that a black "footer" is added. When the website runs on computers, the footer consists only of my email. No colors, no icons. This is another stylistic choice I've made to make it clear which elements are not part of the main features of the site.

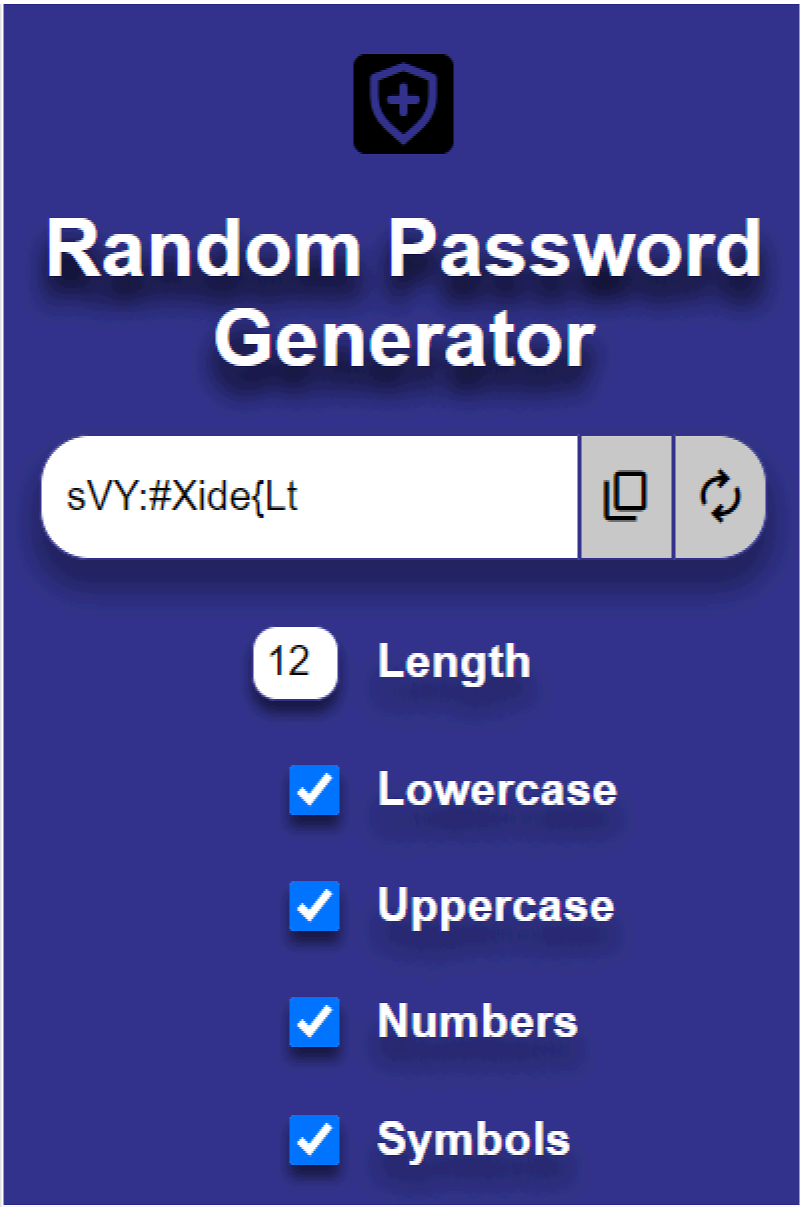
The presence of a black-colored footer on wide devices isn't necessary as I thought the wide-enough margins will make it clear that it is not part of the key-features of the site. Meaning that people would already know that this is the footer.

These changes are what I've learned from visiting websites, in that they tend to have these colored footers presumably for tablets and phones, so that the user knows that this is the bottom of their webpage.

A notable thing is that when the user clicks on the clipboard icon, a text is shown that the code has been copied to the clipboard. This is an important feedback for the user to let them know that they've copied the password to their clipboard. The text then goes back to the generated password.

### Screenshots

320x480:



Random Password Generator

sVY:#Xide{Lt

12 Length

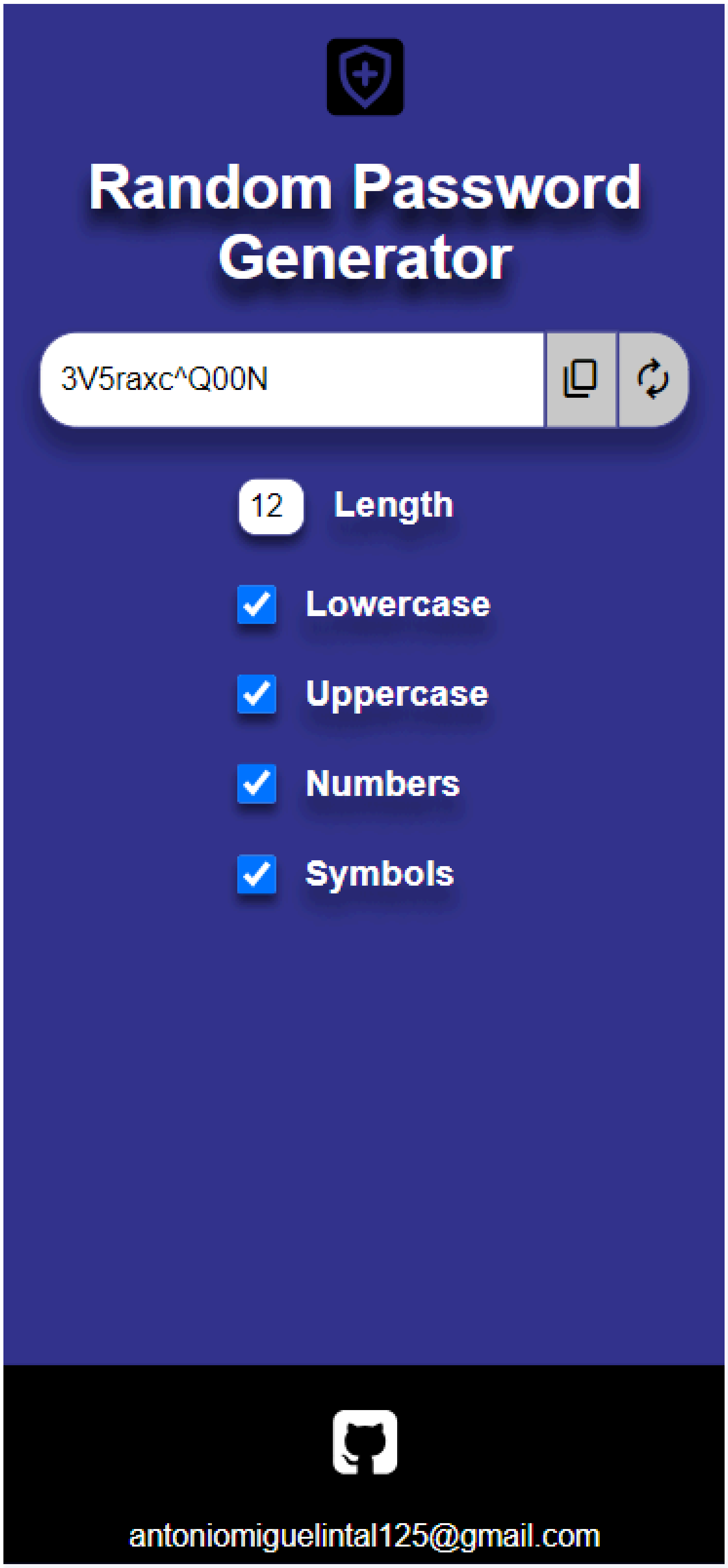
☑ Lowercase

☑ Uppercase

☑ Numbers

☑ Symbols

375x812:



# Random Password Generator

3V5raxc^Q00N

12 Length

☑ Lowercase

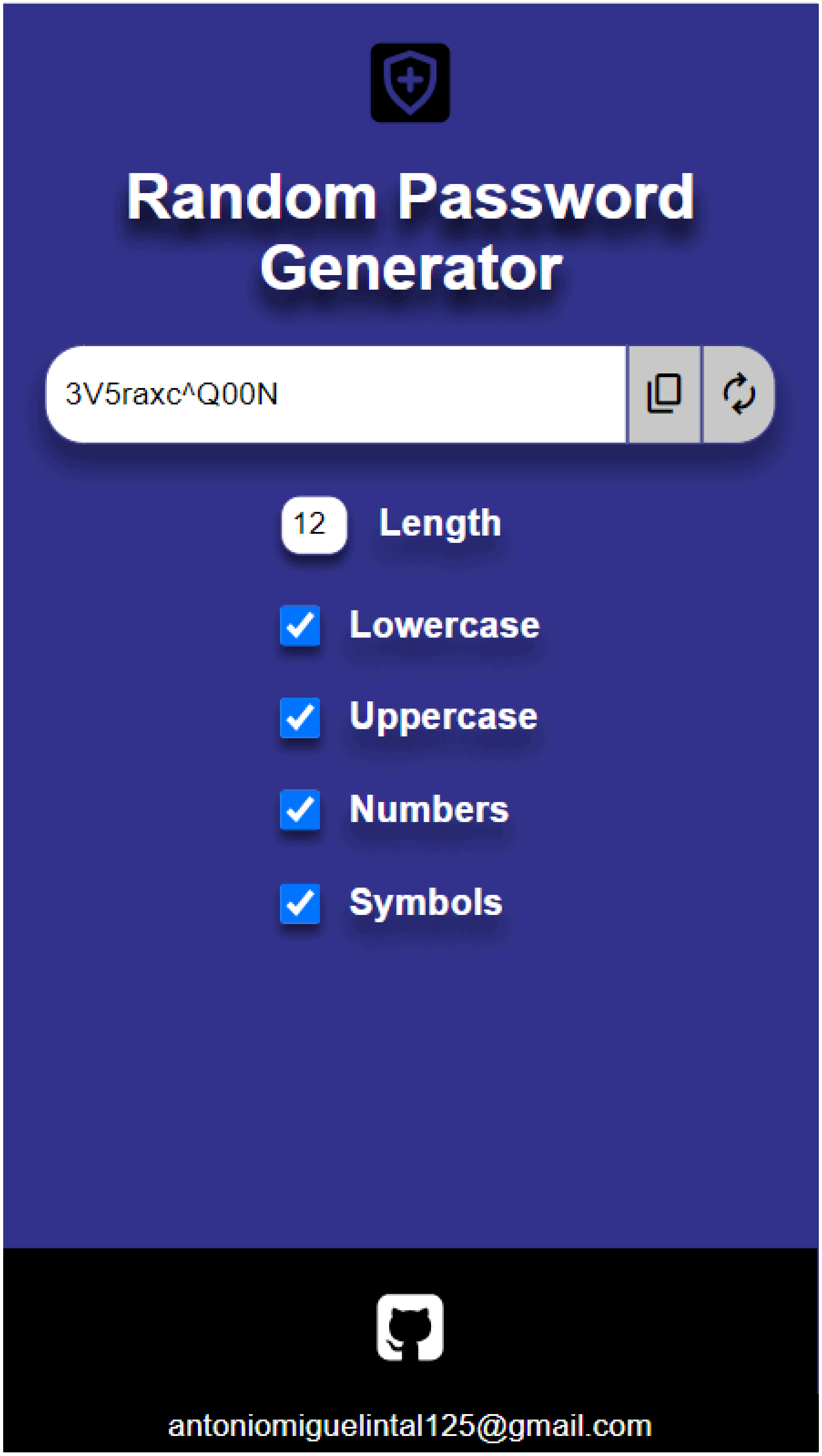
☑ Uppercase

☑ Numbers

☑ Symbols

antoniomiguelintal125@gmail.com

411x731:



# Random Password Generator

3V5raxc^Q00N

12 Length

☑ Lowercase

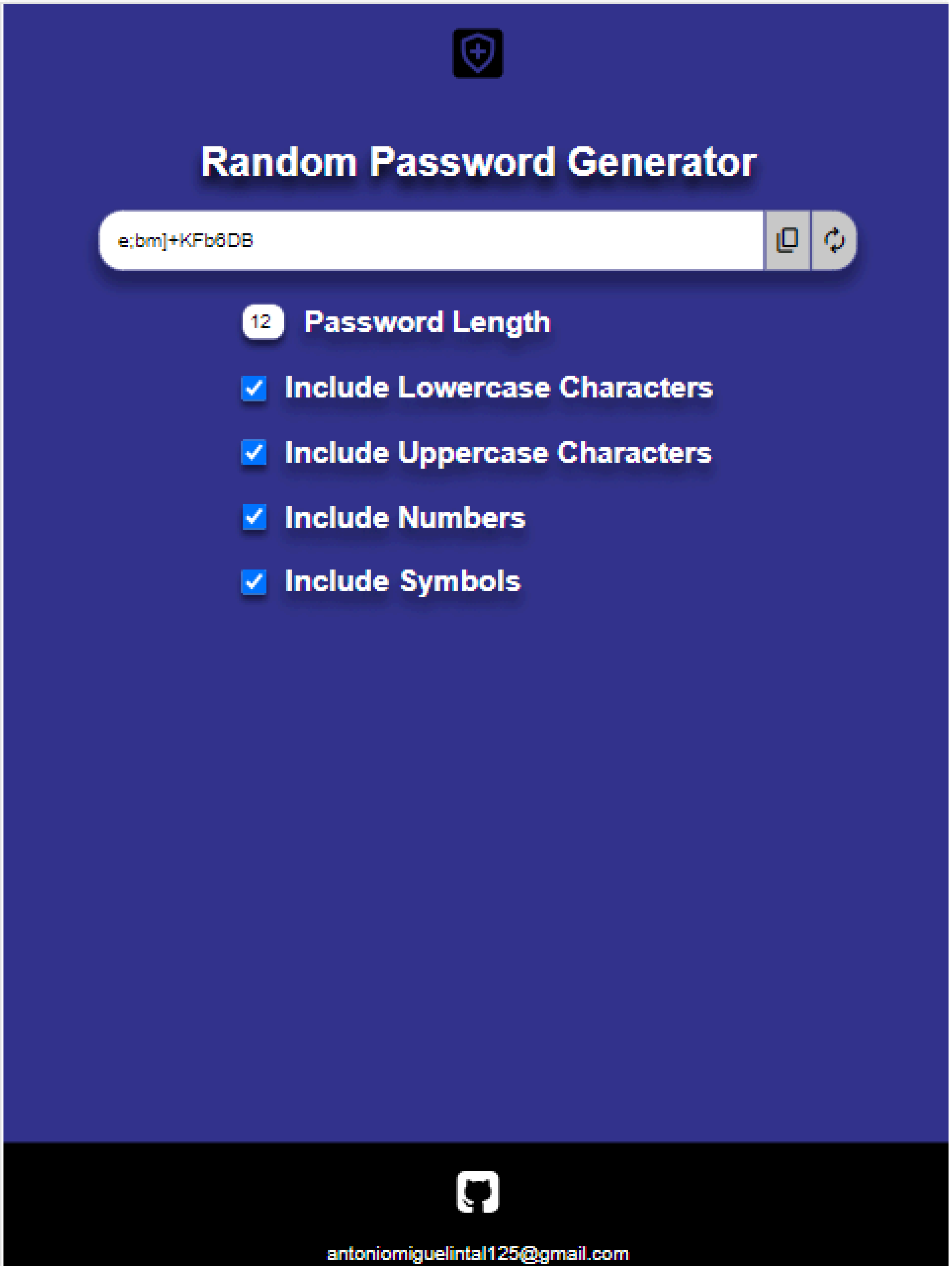
☑ Uppercase

☑ Numbers

☑ Symbols

antoniomiguelintal125@gmail.com

768x1024:



Random Password Generator

e;bm]+KFb6DB

12 Password Length

☑ Include Lowercase Characters

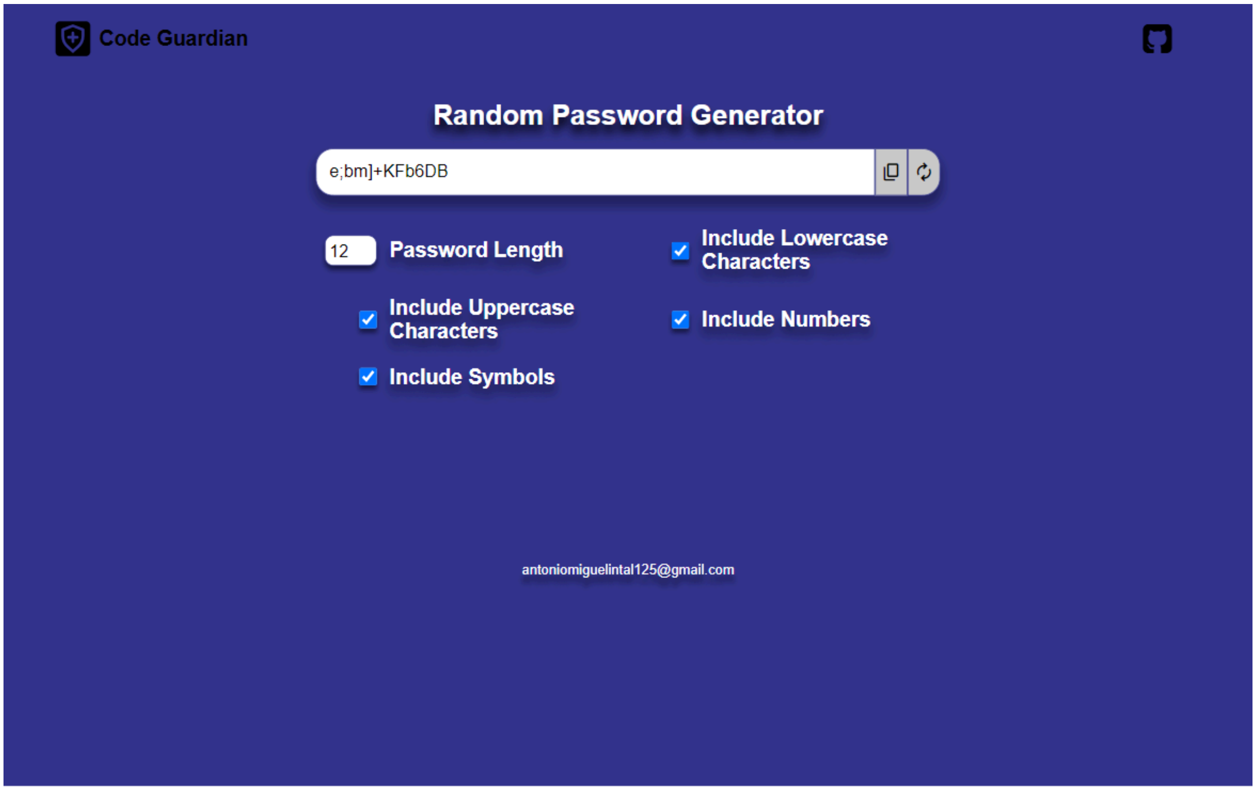
☑ Include Uppercase Characters

☑ Include Numbers

☑ Include Symbols

antoniomiguelintal125@gmail.com

1440x900px:



Code Guardian

Random Password Generator

e;bm]+KFb6DB

12 Password Length

☑ Include Lowercase Characters

☑ Include Uppercase Characters

☑ Include Numbers

☑ Include Symbols

antoniomiguelintal125@gmail.com

## Technical Description

### Random Password Generator

The website uses a self-defined function “generatePassword” built from scratch that essentially creates a “random” string of characters. Whether or not these characters are included in the string are dependent on the “checked” attribute of the HTML buttons. The amount of characters the password can generate is limited to the number given as the length of the password. The function then returns this value.

How this works is that in JS, you will create a variable that contains the document object i.e. a variable that stores the button. Then you will create another variable that stores the “checked” state of the button. This value is inherently True or False.

If the state of the button is True, it will include a set of characters in that category. If False, it will not include.

In the case of the input field to write the length of the password, all you have to do is to limit the amount of characters generated to the given number. What this means is that the password length tells the function: “Hey, you can only generate a password that is *n* characters long. No more, no less.”

When the website first loads, the function to generate a password is already called. This implies that the document objects have already been created before calling to generate the password. The result of this “random” string is then put into the “value” attribute of the input element.

### Regenerating Passwords

In the website, whenever the user clicks any of the options to customize the set of characters the generator is allowed to create, a green line shows for a brief moment. Again, this only serves as feedback to the user to tell them: “Hey, the password has changed to something else.”.

Whenever the user clicks any of the options or the “regenerate” icon, or when the user changes the length of the password, it’s coded that this will call generatePassword to return a new set of characters with the new arguments.

I’ve chosen to code it this way so that the user instantly gets feedback that a new password with the options they’ve clicked has generated, and this is done without the need for clicking the “regenerate” icon.

## Critical Reflection

### Success

I think I’ve personally done well working with what I’ve been given. The website is more or less a reflection of the quality of the course throughout the semester. Whether or not the website looks sloppy and basic, is a topic for another time.

I think my idea of a “password generator site” has been executed sufficiently through this webpage. I wanted something to generate a set of characters, borrowing from preexisting sets of characters, taking into account the eligibility of these preexisting sets to be borrowed or not, and limiting the amount of newly generated characters to a given number.

This is essentially the idea for this site and to reflect that idea through what I’ve learned in HTML, CSS, and JS. It isn’t too much but it isn’t too little, either. It is essentially a: “You get what you pay for” kind of deal.

I said I was going to make a password generator site and I think this contains all the requirements for that website, and slightly more.

The website is pretty good, I think I’ve used some general web design techniques like adding features for user feedback, and updating the layout of the website depending on the size of the screen.

## Limits

### Design

The website is far from perfect, I think more customizations can be done for the site in terms of CSS, at least.

If I’m thinking of design limits, maybe the checkbuttons could be improved as well. They are too small to click on and don’t look good. But the size issue doesn’t affect the quality of the site as users are able to click the description as well, so they are not limited to clicking only the checkbuttons.

I will admit that there are a lot of things I haven't learned about JS when I started this project. I believe this to be the reason why the website remains to be rudimentary. Had I been taught more about JS in school, this website would've been changed for the better.

### Feature

Another thing that lacks in this website is a feature to visibly see the entire password string. I think adding a scrollbar for it would definitely improve the quality of the site, and that, too, would be a unique feature to the site as well.

## Recommendations

Implementing a solution for the mentioned issues above would definitely help the site become easier to use.

Other than that, simply adding more features will do more harm than good to the site as it already escapes the chosen scope of the site, and that is to be a password generator site.

There were a lot of things that were taught in class, but not enough to create a modern type of site for me. I would've looked forward to learning more about JS had we only had more time left through the semester.

It is then concluded that the only aspect that requires updates is the layout of the site. And perhaps clean the source code.

## References:

1. ^ "Norton Password Generator." *My Norton*, https://my.norton.com/extspa/passwordmanager?path=pwd-gen. Accessed 5 January 2024.
2. ^ "Password Generator." *LastPass*, https://www.lastpass.com/features/password-generator. Accessed 5 January 2024.
3. ^ "Random Password Generator | Create Strong Passwords." *Avast*, https://www.avast.com/en-ae/random-password-generator. Accessed 5 January 2024.

***Technical Description:*** A clear description of the programming techniques that you have used to implement your multi-device application. Particular attention should be paid to the use of responsive web design programming methods (~350 words).

***Critical Reflection****:* A critical evaluation of the successes and limitations of your design. This should also describe what personal learning you need to pursue in future to refine your skills (~300 words).