My work process for this exam started pretty well. I started with the HTML code, and feel pretty confident with setting up the html code. Using the project as inspiration for the layout and flexbox uses, I tried my best at adding sections and DIVs where I felt they would be needed. Some were still added later on and others moved. I started by only having one css file, which I later on realized I shouldn't have and hope to never make the same mistake again. Css files become very long very quickly, and being able to separate the content into different files made my corrective work on the panel menu a LOT easier.

I made the panel menu early on, following a tutorial that I have linked in my README.md file, and it helped me create a menu, and a simple Javascript code for it.

I always try to break a big task into manageable pieces, and so I made a test file named testPartOne.html to be able to test snippets of code. I've kept this up while testing all my code, and then testing the complete document later on, to make sure things work when it's also put together. Had I already then implemented a workflow that separated my css code into different files, I think my testing would have been even better and improved the end result. I can sometimes be impatient, and so having to save the css file I just made a small change in, to then be allowed to rerun the debugger to check my edit can sometimes be a little annoying, but it is a workflow I got into after some time.

MAP API

One of my biggest challenges was the map function. For most of the time while I worked on other elements on the page, I had nothing there besides an empty div in the html. In the end I had to make something for it, so I started the process by using google and look for tutorials on how to make the code. Most search results focused on getting the Google Map API Key, but I did end up finding some good help.
I decided to make this part of the Javascript for the map, to test making placeholder information.

```
function showMapPlaceholder() {
  const mapDiv = document.getElementById('map');
  mapDiv.innerHTML = `
    <img src="mapPin.png"
        alt="Map unavailable"
        class="mapPlaceholder" />  `;
}
```

To make the task of creating the map easier for me, it currently only has one map marker, placed on the same farm used in the project file on Figma.

Panel Menu

Even though the tutorial linked in my External resources was of great help with the initial setup, the setup created an element that didn't really want to be placed in the top left corner. This became my rescue, allowing me to specify a locked area where it could also be placed over the NAV bar.

- position: absolute;

To then decide the order of what elements would be placed over other elements, z-index allowed me to adjust order as wanted. I probably did not need to go to such a high number, but for the few elements I needed to order, I aimed high and could add increments of one whenever a different element had placement priority.

- z-index: 1103;

Pixel Count

During my time working on this exam, I've had the PDF easily available to add padding, margin, size and other elements, doing my best to adhere to the size requirements. Some of the measurements provided do not give the result showcased in the PDF. In the This is how it works section, it is supposed to be 785 px in height, but the four elements on the right side, each require 197 pixels in height. 197 x 4 is 788 pixels, and with more time, I would like to change this so things line up correctly. I tried changing the pixel height from 785 to 788, but that did not solve the problem.

ChatBot page vs Window

I depended a little too much on only the PDF file for most of my time working on this project. When it then came to the chatbot, I made it a new html page. Studying the Figma file later on in the week, I realized this was wrong and so I hope to set it up as a window over the other pages instead of the page that it currently is.

I shared my repository in my study group, where I have gotten some feedback that I will take into consideration and hopefully implement before the project is due to be handed in. Looking at other student's code, and file structure also helped inspire me when it came time for me to work on my README.md file. I have also tried to be good with giving myself breaks so that I can return to inspect my code with fresh eyes. The placeholder for maps was an idea that hit me while I was taking a break.

Being able to return to my code, 10 minutes later or the next day, makes me more confident that it can be read by others. There are still elements I would like to clean up, but being able to look at the code I have written during this exam, I am very happy for everything I have learned, and I still have a lot to learn.

Implementing and relying on AI instead of humans, means that jobs across many sectors are also threatened. Having a fully AI driven system for helping customers runs the risk of running into instances where the AI struggles to comprehend what a customer wants, whereas a human can process and help with elements a programmer hadn't even thought to implement in their AI training data. The AI is only as good as its guidelines and restrictions, meaning if it has been trained on data that includes biases, or has no filter settings made, the prompts returned from the AI, can contain unwanted results.

More and more, AI is implemented to be used in risk assessment, customer service and tested for cars. With this autonomous system in place, a concern regarding responsibility should be considered. If an AI driven system, like self driving cars, causes harm, should the car owner be held responsible, the programmer who wrote the AI or the car company that implemented the use of AI. How can a half conscious string of data be held responsible when humans are harmed?