|  |  |
| --- | --- |
| Name | Animal Care |
| Creators | Tamás Visegrádi, Gábor Kelemen |
| Project Type | School project |
| Project length | Seven days |

**Frontend**

**HTML:**

On the first line, we can see a DOCTYPE element which specifies the document type. The HTML tag specifies the document language. Then there goes the head which contains three meta tags, the title and some other cdn script for ionicons, for the favicon and also for Tailwind CSS which is one of the most advanced frameworks out there.

Inside the body tag, we have the offcanvas which basically gets activated when the user clicks on the profile button that can be found on the bottom navigation bar, among other options such as: home, github and contact. Inside the offcanvas, the user has the option to check all the item inside the basket, sign in and out in case the user still hasnt made an account yet, and to log out if the user already logged in at some point. Furthermore, here is where you can toggle the dark/light mode, and also to change the language. The user can choose from eight options. By clicking on the contact button, a modal popup will appear, where all of our email contacts are available.

The next one is a preloader for the website. On the top of the page, we can also see a progress bar which detects scrolling thanks to Javascript. After the progresss bar, we can see the navigation bar with its own respective header semantic element. Inside this element, we can see another semantic one, which is called the nav. All the links are inside a unordered list, each redirecting to its distributive part of the home website.

After this we can see the hero section where the user can read the website title and also an abridged description. There is an about us section where we basically introduce our website and what we are aiming for. Here, we also delve deep into the history of our company and how we gained and augmented our prominence.

We also offer services, namely three: veterinary, grooming and pet boarding. The next is the our products section where we used Tailwind CSS cards to projects our latest and most sold products for animals. All these sections and items inside our website, have their own individual classes and ids if need be. By clicking on the ’Read more’, the user can visit a separate website to gain valuable information and insights regarding the product they are looking for, anticipating to buy.

For the ’Our Brand Deals’ section, we used an automated image slider/carousel, to show the logo of every brand we have a contract with. These logos have the same classes.

Now for the contact section of the page, we used numerous label and input tags inside the form tag. These labels have a for value, a name and id. The same goes for the input tags, except for the ’for’ value, and it has an additional value, which defines the type of the input.

Last but not least, we have the footer at the bottom of the page and for that, we used the oh so called footer tag to get the job done. This footer contains the basic contact information for our company. Thats where you can reach us and expect a respond the most quickly. Moreover, you can find the company’s Instagram, Facebook, Twitter, Github and Dribble account. We desvided all the specific elements inside the footer using divs, but we also used a section tag at the beginning which is useful for creating spaces between websites contents.

Lastly, you can see the code for the bottom navigation mentioned earlier, which is also a preset Taiilwind component.

**CSS:**

Inside the CSS file, you can see the :root selector, where we basically defined a few preset values, such as colors and some sizes in pixels. Inside our file here, we have got a plethora amount of selectors. The .dark specified selector, allows another selector to apply specific styles when the content background is black. In this way, we were able to make our website have a dark mode feature. with the ::selection selector, we were able to specificy the colors of the highlighter.

For most sections and items, we used grid technology among other ones, such as flex or flexbox. We endeavored beefing up our website the most, by being creative, quixotic on the types of selectors we use. We knew when to use ::after and ::before without major gaffes. We know the grid, flex technologies well, so we know how to use the horizontal, vertical align properties, among with the flex-view,flex-direction but even the grid division didnt cause us any ordeals.

We use a few @keyframes to animate our website more efficiently. We focused a lot on finding the apt, consummate colors for styling and overall look. We used @media queries in order to make our website fully responsive. Our website is responsive ranging from 1700 pixels to 300 pixels, but we still havent made it responsive for bigger screens (1440p,4K)

At specific parts of the css code, we resorted to using the !important specified selector to override some already defined values at some parts of the code. We used max-width and min-width as parameters for the @media queries.

**Backend**

**JavaScript**

Inside our Java Script file, first we can see a function called changeLanguage(). Inside this function, we choose our current lang’s Json file which is already inside a folder. The by using Fetch() we first check whether the response was ok, after that we change the language of every single website element by extracting all the translated string values from our Json database. If the response fails or the database doesnt exist, the program throws an error to the user.

After this, we can see a few const which store the navigation links and also the hamburger icon (which is a button of course), and an addEventListener() which is for being able to toggle the hamburger button.

The second function is called updateFlag(). Here we select the dropdown which contains all the language options that is available (that is: english by default, italian, arabic, russian, german, spanish, french and english once again as an option) By using an array, we store all the ids of texts or other contects, that we need to change it font size due to the variety of different language texts. We do that by using the forEach() method to iterate over the elements of this array.

We use a switch which checks what the value of the dropdown is (so which language name) and according to that, it replaces the flag with its correct, matching one. The next part of the Java Script file is the automatic slideshow. Here, we use an arrow function.

We make a variable for indexing, a const for collecting all the logos inside the .track div, and another const for storing the .track div class. We define the index by adding one, dividing that value with the length of the logo carousel,then getting the remainder of it.

Then we make a const variable where we multiply the width of the logo with this index and according to this value, we move the entire logo carousel on the X axis. On the next const, by using the SetInterval() function, we give the function name mentioned above plus the amount of time it should call the function on repeat in milliseconds.

The next is where we toggle the dark mode on and off. We select the toggle button and the icon and we apply an event to this toggle where we activate every single selector that has a .dark and apply the styling inside their blocks on the entire website. Furthermore, we change the icon according to the toggle button’s state: light – sun, dark – moon

The next part is where we set the preloader of our website. We apply an onload property to the window module to load the preloader after the entire window has managed to load. Then we store all the divs and contents inside our preloader using const variables. By using the setTimeout function, we set the display to none so that it disappears, and make the content of the webpage appear. By setting the time out to 3000 milliseconds, it means that it takes three seconds for the preloader to disappear and for the contect to be accessible.

The next fraction of the code uses the onscroll property of the window function, that gets activated when the user scrolls. Here we store the document, the window height and how the height changes when we scroll either up or down. We convert it to a percentage and by that, we can basically measure where our progress bar actually is,and given this value,we change the width of the progress bar.

Moving on, we have the offcanvas where we store the closing,opening button, the content and the head div itself. We apply listeners to these buttons which makes the offcanvas appear/disappears using show and hidden classes. We can also see the modal part of the code where again we store every single intrinsic values, contents and divs using const variables. We again use an event for the opening and closing buttons of the modal. There is also a third listener which ensures that in case you dont click the closing button, but instead click outside of the modal, it still closes it.

Last part of the code is the functionality of the dropdown. This is for the language selection. The selectLanguage() function makes sure that the language name next to the flag is the correct one.

And of course we have a last listener, where we make sure the dropdown actually appears and it is toggled. If the parameter value that is inside the listener is not part of the dropdown content, then no flags or language name will appear after clicking. And by all means, the translation goes pear-shaped.

**JSON**

We have seven Json files in total. Each one with their own respective language translations. We selected all the ids and classes of the site’s contents, and define them as Json keys so we can access them easily and translate the specific text into the specified language.

Table of Contents

Frontend1

HTML2

CSS3

Backend4

JavaScript5

JSON6