Noroff
School of technology
and digital media

Semester Project 2

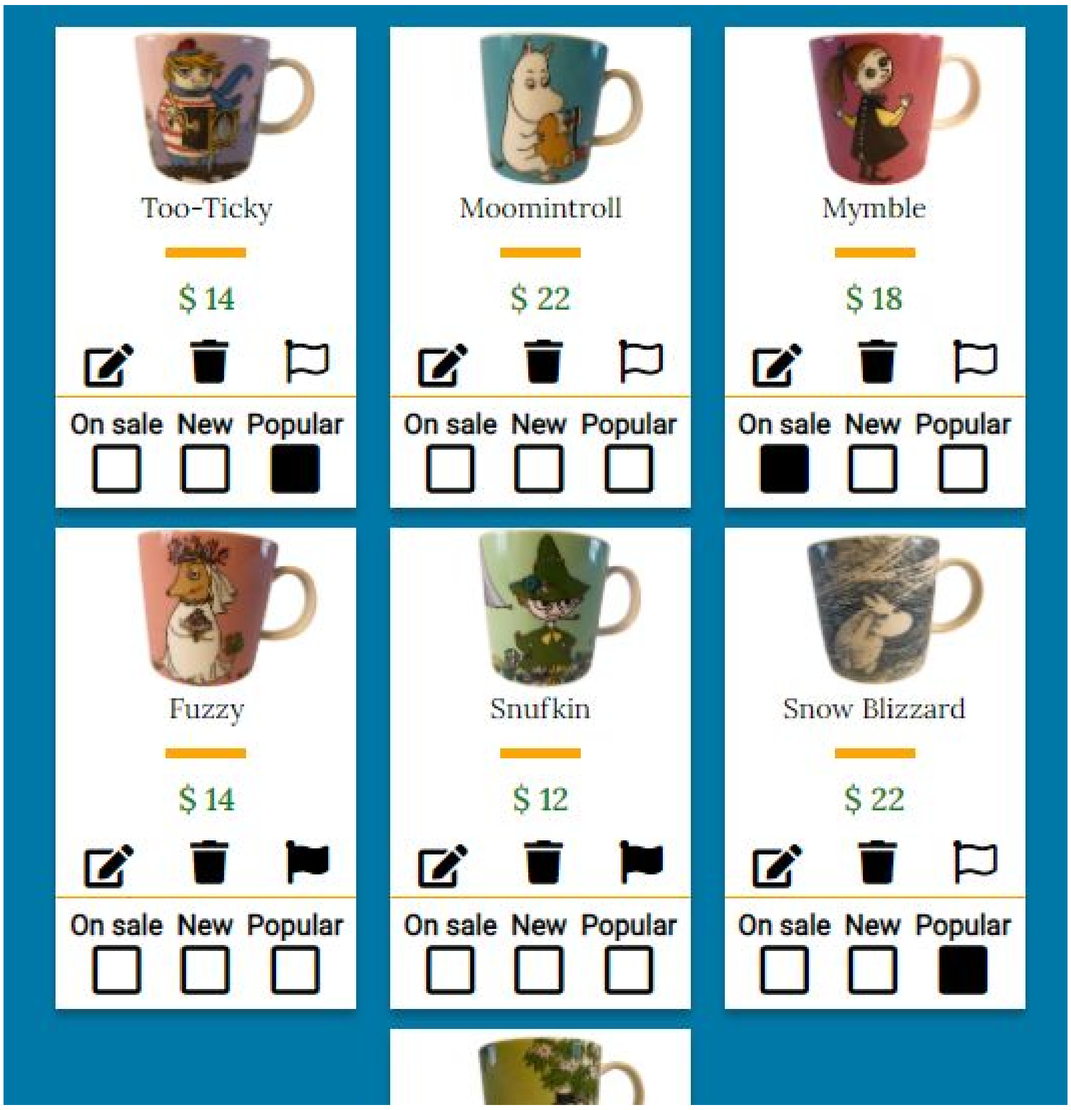
Trond Fuglseth Spjelkavik

Design

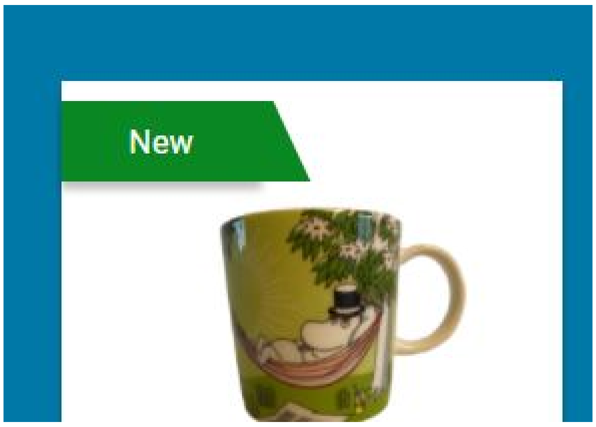
What went well on the project

The design was intended for users who are interested in Moomin Cups. And the target audience is from age 18 - 40. The layout and design is easy to navigate, with clear colors and call to action. I made the design and layout on Figma before I started with the code.

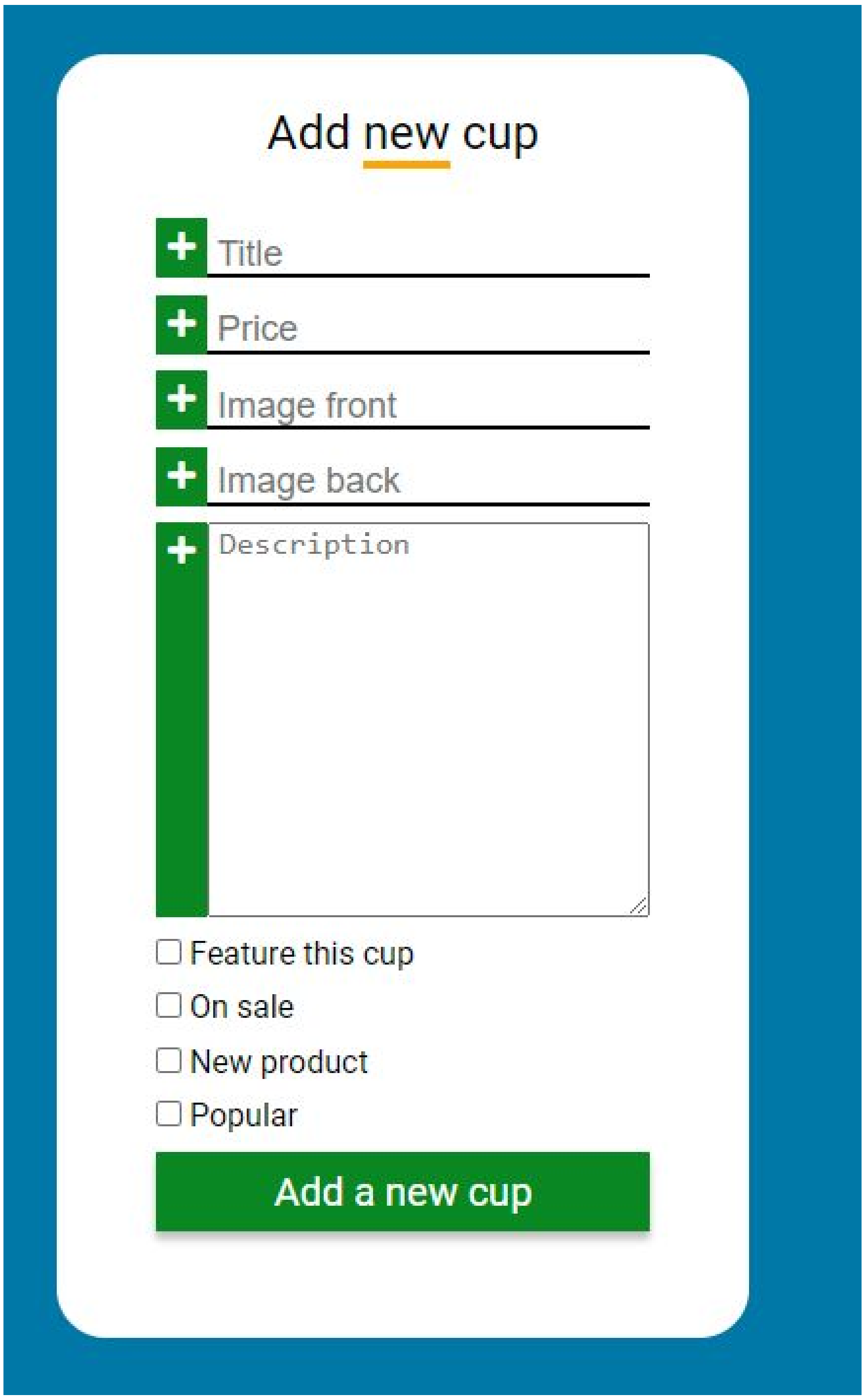
I took all the product images on a mobile camera, and uploaded them to my own domain, and linked the url in the API. I also made an easy to use UI for the admin users, where they can change, add and edit the cups.



Here the admin user can flag a featured cup, delete and edit a cup with a click on the icon. I also added an option to choose between “On sale”, “New”, and “Popular”. This will change the element from false to true, and true to false and display the new feature on products and/or homepage.



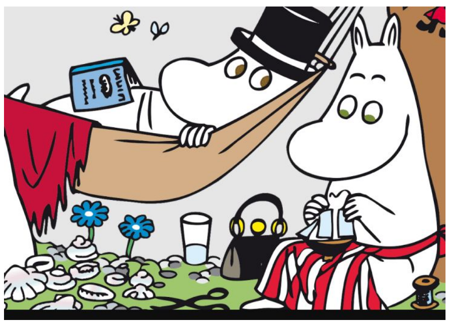
To add a new product, i created an form that let the user add a new cup, and see the changes when the admin users clicks submit:



The admin user also has the choice to add features to the product.

On hero and moomin pictures I added a background(green) so that the characters are seen to be sitting in grass. This was created with a <div>.

Original vs Current



What was difficult/didn’t go well on the project

I still need to practice CSS and Design, and be more comfortable with design. I did not use any bootstrap because I wanted to learn more about css and sass.

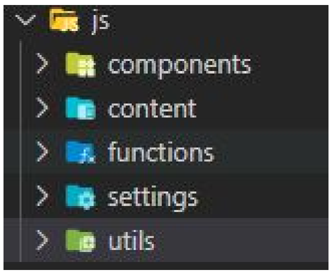
What would you do differently next time

I will continue to learn and practice design and css. Next time I will use more time on the prototype and design, and finish the prototype with all interactions before I start with code.

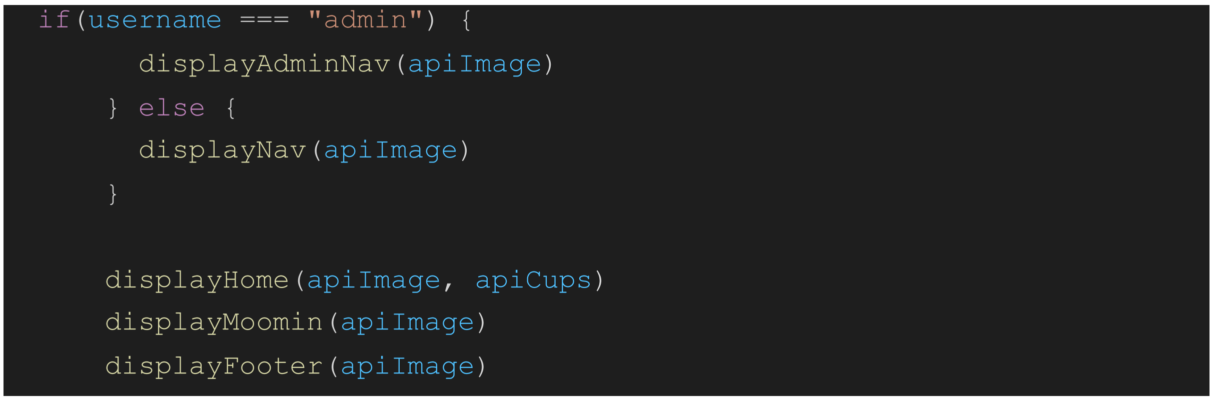
Technical

What went well on the project

This project was a great learning experience, and fun to code. I made my own API using Strapi and Heroku. The images in the API are external urls, this is because of the filesystem heroku uses, and the uploaded images get deleted after dyno restart. My setup for the JavaScript:



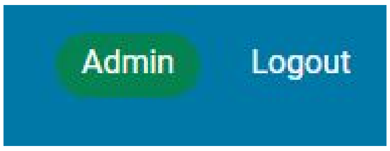
Components are where I do an API call, and import the javascript functions to display html with other functions. Snippet from home.js:



```
 if(username === "admin") {
        displayAdminNav(apiImage)
      } else {
        displayNav(apiImage)
      }

      displayHome(apiImage, apiCups)
      displayMoomin(apiImage)
      displayFooter(apiImage)
```

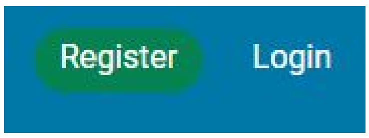
When the pages loads it checks if the username is Admin, and if it returns true, the admin get an nav for admins:

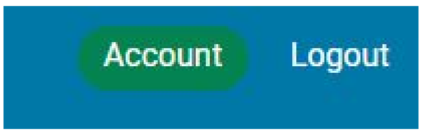


And can navigate through the site, and quickly get back to the admin panel.

If it's not an admin, it get either a nav for logged in user og just guests:

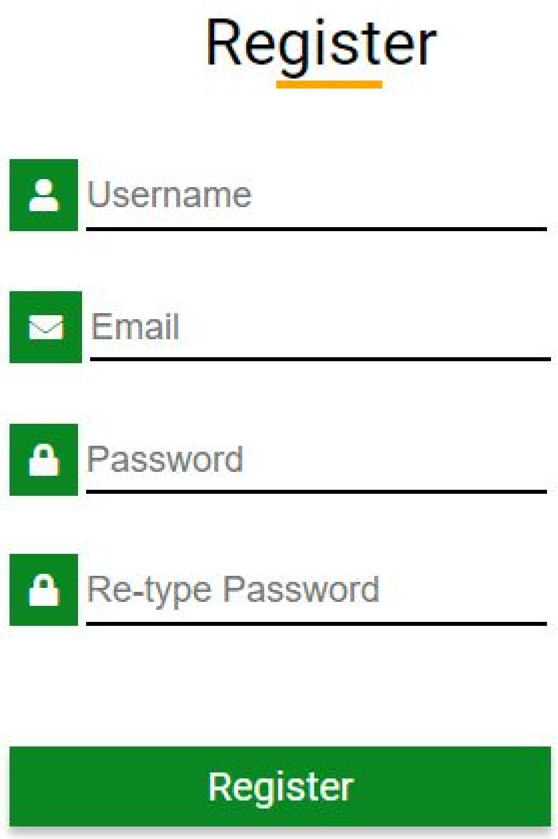
<u>Guest vs Logged in</u>



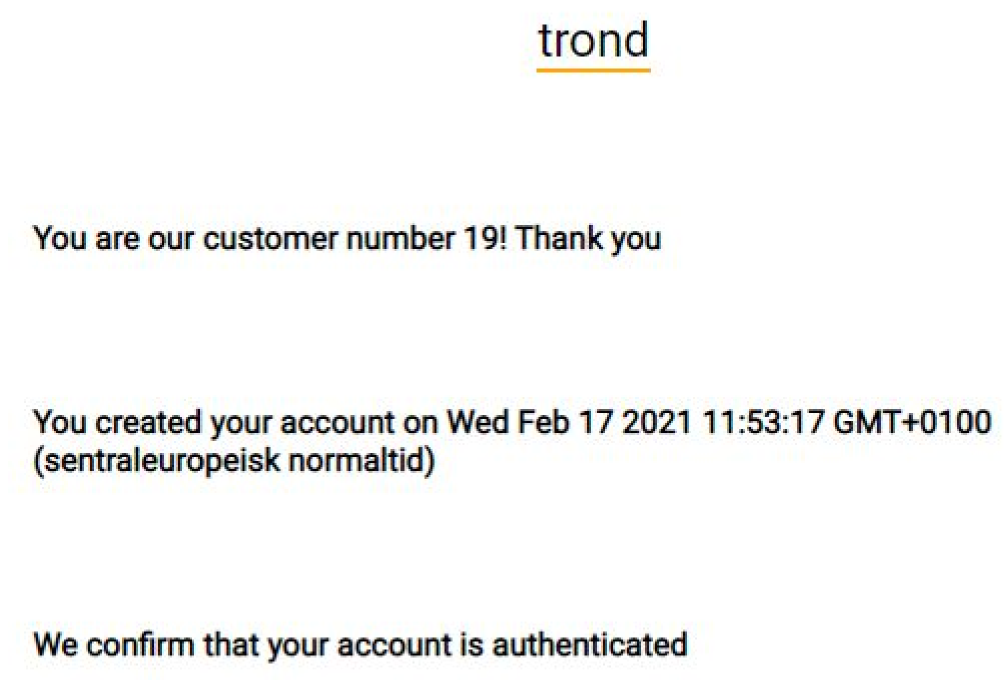


I wanted to learn more about API and registering new users. I created a form that let a new user register to the site. When the user registers to the site, the API updates and saves the credentials.

<u>Register:</u>



Account Page:


trond

You are our customer number 19! Thank you

You created your account on Wed Feb 17 2021 11:53:17 GMT+0100 (sentraleuropeisk normaltid)

We confirm that your account is authenticated

This was an extra feature I wanted to learn, and see how the API works with new users.

Content

The content folder holds all the html and interaction on the page.
This is also were I check if the user is logged in

displayNav.js:

```
const loggedIn = getUsername();
```

if the user is logged inn it either display a nav for admin, or logged in users:

```
if(loggedIn) {
        navContainer.innerHTML +=`
```

I also created 3 variables to check if the products is on sale, new or popular:

```
 let onsale = "";
 let newProduct = "";
 let popular = ""
```

```
 if(cups.onSale) {
          onsale = `<div class="feature-style"><p
class="feature-style-content">On sale</p></div>`
        }
        if(cups.newInStore) {
          onsale = `<div class="feature-style"><p
class="feature-style-content">New </p></div>`
        }
        if(cups.popular) {
          onsale = `<div class="feature-style"><p
class="feature-style-content">Popular</p></div>`
        }
```

On displayHome.js i check if the products have the feature flag on: If the product has feature flag on, then create html:

```
if(cups.isFeatured) {
            featuredContainer.innerHTML +=`
```

To let the user see the front, and back image of the product, i created an Array with the pictures, and a if statement that checks the .src on click.

```
let imgArray = [ apiCups.image_back, apiCups.image_front]
let img = 0;

detailsImg.forEach(btn => {
  btn.addEventListener("click", () => {
    detailsMainImg.src = imgArray[img]

   if(img === 1) {
     img = 0;
   } else {
     img++
   }
```

On the cart page, I use localStorage to check if the user has selected a product, and to sum the price I created an empty array and a function. When the html is created it pushes the price into the array:

```
sumPrice.push(cup.price);
```

I check the total, and round the number.

```
function getSum(total, num) {
    return total +  Math.round(num)
}
```

To get the total price i use .reduce()

```
const totalPriceAmount = sumPrice.reduce(getSum, 0)
```

If totalPriceAmount is true, then I create a html:

```
if(totalPriceAmount) {
  totalPrice.innerHTML += `
  <div class="total-price">Total price: $
<strong>${totalPriceAmount}</strong></div>
  `
}
```

<u>Functions</u>

In the functions folder I have 2 functions. One that removes cart items from localStorage, and one that displaysMessage on form validation.

removeCartItem():

```
export function removeCartItem(e) {

    const id = e.target.dataset.id

    const currentCartItems = getCurrentChart();

    const newCart = currentCartItems.filter((cup) => cup.id !== id);
    saveChart(newCart)

  function saveChart(cup) {
    localStorage.setItem("Inside Chart:", JSON.stringify(cup))
  }

  location.reload();

  }
```

Settings

In the settings folder i have a api.js, with the apiUrls:

```
export const baseUrl = "https://mummi-api.herokuapp.com"
export const imageUrl = baseUrl + "/assets"
export const cupsUrl = baseUrl + "/moomin-cups"
```

In the api I use two collection types. One with the products, and One with the images.

Utils

In the utils folder i have 3 files.

searchCups.js

Here I use an onkeyup, and filter the cups with an onkeyup event.

```
 const searchValue = e.target.value.trim().toLowerCase();

    const filterCups = cups.filter((cup) => {

      let cupName = cup.title.toLowerCase().includes(searchValue);

      if (cupName) {
        return true;
      }
```

The two remaining files are for localStorage. One to retrieve the cart items and the current feature products.

```
export function getCurrentChart() {
    const favorites = localStorage.getItem("Inside Chart:");

    if (!favorites) {
      return [];
    } else {
      return JSON.parse(favorites);
    }
  }



  export function getCurrentFeatured() {
    const featured = localStorage.getItem("Featured Product:");

    if (!featured) {
      return [];
    } else {
      return JSON.parse(featured);
    }
  }
```

The userStorage.js gets and removes items from the localStorage.

I also have the logout function, and remove the token and user when logging out:

```
export function removeUser(user) {
  removeUserFromStorage(userKey, user)
}

export function removeToken(token) {
  removeUserFromStorage(tokenKey, token)
}


export function Logout() {
  removeToken();
  removeUser();
  location.href = `index.html`;
}
```

What was difficult/didn’t go well on the project

Regarding the code I learned alot about API, and working with the API. I enjoyed the assignment, and had some problems with the checkbox value. The checkbox value returned “on”, and I needed the value to return true. To fix this I needed to change the value on all products to false, and then when a user clicks it gets true. I also need to practice more form validation.

What would you do differently next time

Next time I would create an API from scratch and learn more about auth, and how to create an API without a CMS. I also would use more time to plan the file structure of my files. I created a database in excel before i created the API, and I would also create a file structure in excel before i start the code next time.