07.01.01: Static Product Assignment

Description

Jonas' example site: https://kea-alt-del.dk/t7/example-site/

Your task is to build a static website for our clothes shop. In this context, static means the content, images etc. does not change and there's no integration with a database.

During the next two weeks we will make it dynamic, so that all products are fetched from a database, which means that the site updates automatically when the data is changed.

Initially, you'll need three pages (see image below).

A product page for showing the details of a single product

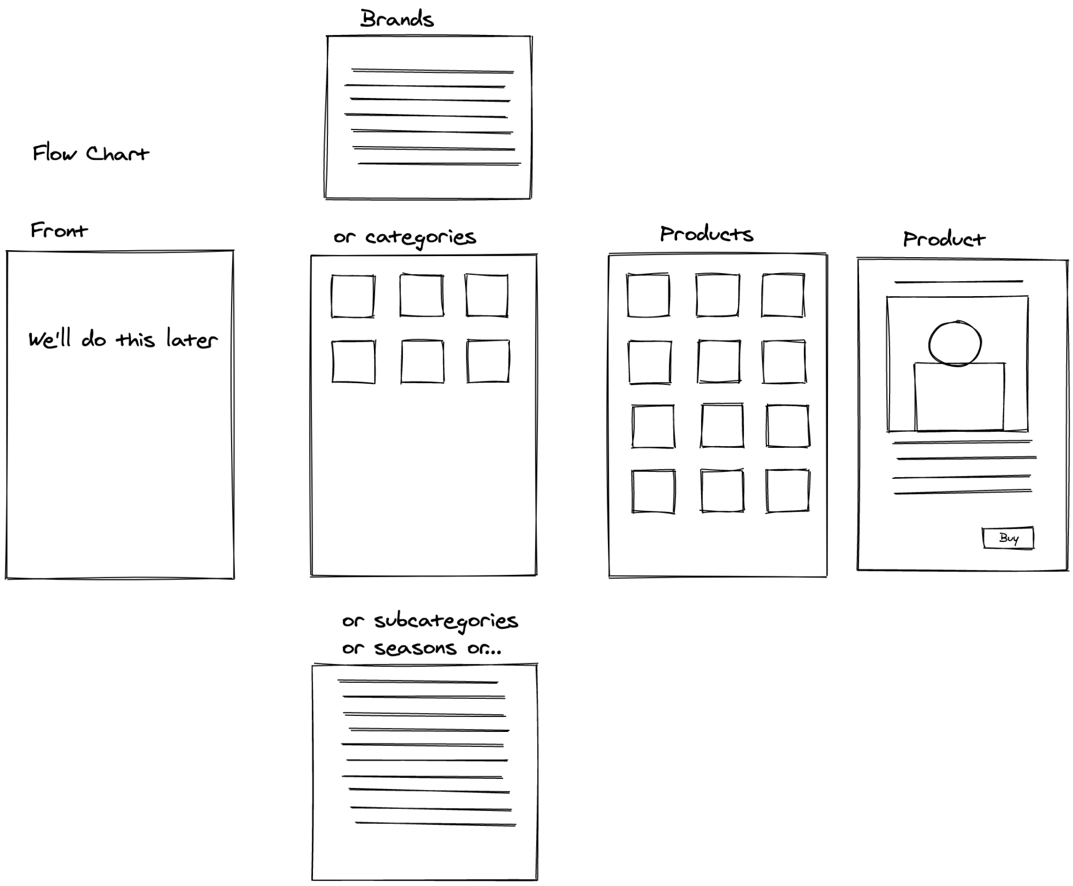
A product list page for showing a list of products

A category page for showing categories, brands, subcategories, seasons or similar

The front page is added as the last thing.

So the flow is something like:

1. The user is on the "category" page and clicks on a category
2. The user enters the "product list" page and sees a list of product from that category
3. When the user clicks a product, they are sent to the "product" page



When designing each page, you'll have to consider what data you have to show. In the following section I'll go through the data, how to access it and a few considerations

The Data

Start by taking a look at some sample data in [this overview] to get an idea of what you can do.

Each of the three pages will have access to different data, let's start with the simplest one.

I recommend that you name each file exactly as I do, that will make following along easier.

For all links that follow, I REALLY recommend you use Firefox, as Firefox knows how to display "JSON" data. But be my guest, try the links in Firefox and Chrome and see the difference.

product.html

Example Data: https://kea-alt-del.dk/t7/api/products/1163
Image (same image, different extensions and resolutions)

- https://kea-alt-del.dk/t7/images/webp/640/1163.webp
- https://kea-alt-del.dk/t7/images/webp/1000/1163.webp

- https://kea-alt-del.dk/t7/images/jpg/640/1163.jpg
- https://kea-alt-del.dk/t7/images/jpg/1000/1163.jpg

Remember, this is just a sample of the data so you have an idea. This single webpage will (soon) be able to show all the 44.000 products that are in the database!

Challenges

Remember to take into account (and present visually) that the product can

- be sold out
- have a discount price

productlist.html

Example Data: https://kea-alt-del.dk/t7/api/products
Images are accessed by taking the product's id and using that instead of the 1163 in the examples above.

This page should show a list of products. Try to be consistent. Create the layout of a single “card”, then copy it and change the data. Stop when you’ve 3-4 products, and copy everything a few times so you know how your site behaves with lots of products.

Each card should link to product.html

Challenges

Remember to take into account (and present visually) that the products can

- be sold out
- have a discount price

category.html

Example Data, choose one of the following:
Categories: https://kea-alt-del.dk/t7/api/categories
Sub Categories: https://kea-alt-del.dk/t7/api/subcategories
Brands: https://kea-alt-del.dk/t7/api/brands
Seasons: https://kea-alt-del.dk/t7/api/seasons

In a perfect world, the user would be able to filter and navigate our site using all of the above, but I think we can agree that the world is not exactly perfect these days. So, pick one and build some sort of list when the user can select something to filter by.
Make all links point to productlist.html

Challenges

- Some of these lists are extremely long, don't use all the data now, copy paste instead once you have your structure.
- How can you present this simple data in an interesting way.

Create your design so that all of these differences can be shown.

Hint: Make a class for each possibility that you add to the product.

UI requirements:

Use CSS grid and optionally flexbox to layout the site and the lists.

Remember, you can have grid inside grid, grid inside flex, flex in flex etc. Having multiple simple grids are WAY easier than one big.

Hints and good ideas

Remember to make your design ready for the dynamic part, make the products as similar as possible and make a mental note of the differences.

Use as many relevant classes as possible, a soldOut class for products that are… sold out, a discounted class for products that are on a discount etc.

Add comments for yourself, for instance if there's an element that should be removed in some circumstances.

Delivery

This iteration is not a hand-in, but rather something you will continue working on after the hand-in.

Have it ready by Thursday morning.

Ressources

- The sample data
- The API

Feedback

We'll see a few, randomly chosen, in class Thursday morning.d