



CHANGEMAKER EDUCATIONS

EXAMENSARBETE

JAMstack Ecommerce

WRITTEN BY

Annie Taylor Chen

WUE18


Jan 16th, 2020

Table of Contents

The Demand of Modern Commerce	3
What about Amazon?	4
The Challenge of Old Platforms	5
What is JAMstack?	6
The Benefits of JAMstack E-commerce	7
Petite and Minimal Case Study	9
Conclusion	17
References	18



The Demand of Modern Commerce



Xiao Li, an accountant, works in fancy CBD in downtown Shanghai. She lives in the suburb and needs to commute every day to work. She likes to browse products on her mobile phone when she is riding the subway, and places order. In the evening when she gets home, her parcel will be delivered to her door.

Anna, a fashion design student, studies at Konstfack in Stockholm. One of her favorite pastime is to browse online for inexpensive but unique items, be it clothes, shoes or accessories so she could buy them, exam them, and gain design inspiration from them. She likes to browse those on her phone when she is commuting or waiting for friends in cafe or restaurant, then she will order everything on her desktop as it's easier to compare prices and have an overview since she has lower budget.

Paul, a business executive from UK, is transiting in the early morning at one of the international airports. He is extremely busy but he doesn't want to forget his wife's birthday. He is browsing online flower shop to book a nice bouquet for her to deliver before noon, then book a fancy dinner for the evening as a surprise gift, as he happens to conclude a business earlier than planned, and his wife doesn't know yet.

What you see here is a common scene in our everyday life today. Life is getting hectic and people are always on the go. The fast-paced modern life gives us less time for ourselves, therefore we expect the convenience and speed from the services we use. Literally, any service. Online or offline. Or both.



What About Amazon?

“**If you do build a great experience, customers tell each other about that. Word of mouth is very powerful.**”

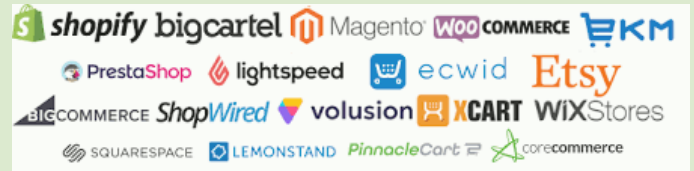
Jeff Bezos,
(Founder of Amazon.com)

No doubt, Amazon has been a giant e-commerce in the scene. And what's more, they have the resources to constantly innovate their service over the years, both online and offline. They have reorganized their working process, digitalized the organization, modernized their warehouse management by implementing robots and they're exploring robots as delivery as well. The products that can be bought on Amazon has been varied, low priced and delivered within a day or two. Feeling threatened huh?

Of course, Amazon provides the majority of nondiscretionary, commoditized portion of the retail market, but the discretionary, specialized products remains for the others to explore and expand. Not every customer is after the cheapest product. Within this field, the most fun, interesting, innovative, high value and emotion-driven purchases would be where you can actually win. And in order to win you need to provide highly inviting, engaging, unique, personal yet authentic experiences for your customers. This is where you can actually do better than Amazon.



The Challenge of Old Platforms



In the past 20 years, most e-commerce solution relied on monolith commerce platforms. They have very rigid architecture, limited choices of themes or layouts, less customization, and can be expensive to modify or maintain. Not to mention they're totally designed to function in one space - the web. If you want to explore other areas, such as AR (augmented reality), IoT (internet of things), social media shopping, mobile shopping, you would have to hire another team to work on another set of solution, which can add to huge increase in budget and time.

Gartner, in one article about 10 trends in digital commerce, lists visual commerce, personalization, unified commerce, thing commerce and API-based commerce as the must-haves for the future. As you can see, none of them is easy to achieve if you stick to old-fashioned traditional e-commerce platforms such as Wordpress with Woocommerce, or Shopify. That's why using JAMstack can be beneficial for the future development of e-commerce.

What is JAMstack?

JAM stack is a modern architecture – fast, secure and dynamic, created with JavaScript, APIs, and pre-rendered Markup. Compared to traditional web architecture, it's served without back-end. (see picture below)

JavaScript

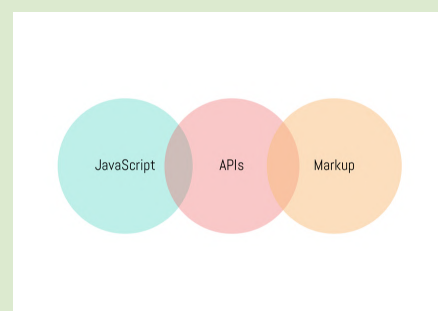
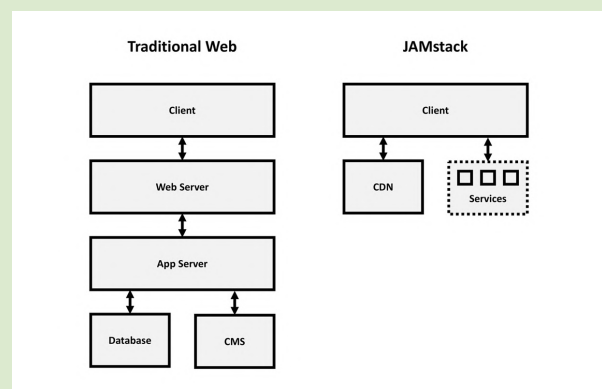
Any dynamic programming during the request/response cycle is handled by JavaScript, running entirely on the client. This could be any front-end framework, library, or even vanilla JavaScript.

APIs

All server-side processes or database actions are abstracted into reusable APIs, accessed over HTTPS with JavaScript. These can be custom-built or leverage third-party services.

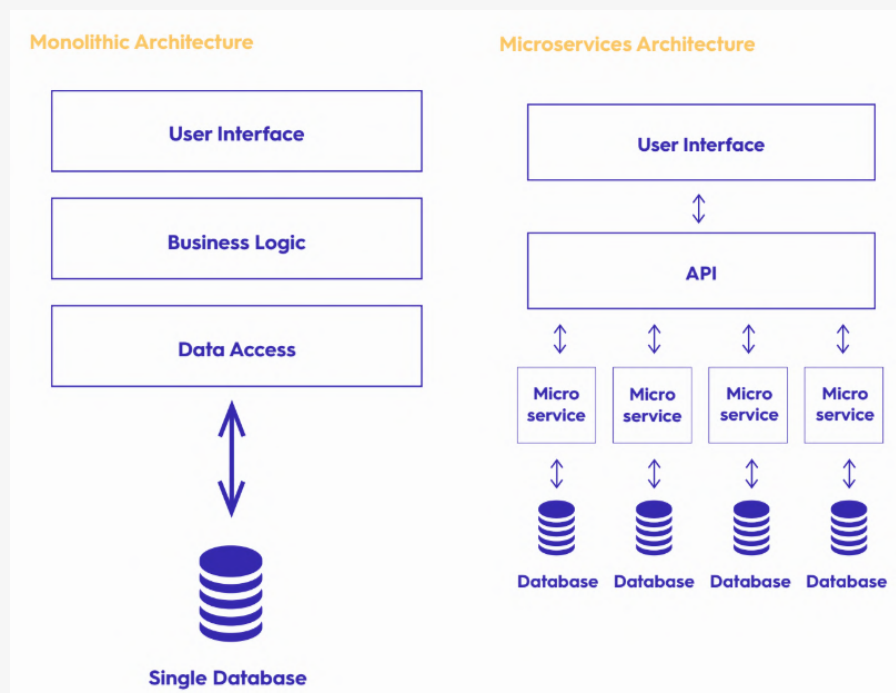
Markup

Templated markup (for non-coder content editors) should be prebuilt at deploy time, usually using a site generator for content sites, or a build tool for web apps.



The Benefits of JAMstack E-commerce

As you can see that from JAMstack, things are decoupled and delegated to the micro-service experts, then integrated into a whole to become our own.



Customization

You are no longer restricted to the limited themes and layout provided by monolithic software, instead, you can use any front-end framework to build a highly customized UI and UX to shape your brand identity, which then improves customer conversion, increases their loyalty, therefore increases the lifetime value of your brand.

Scalability

Since your front-end and back-end are separate, they can grow at different speed, according to the business demands. Even your front-end receives huge traffic, your back-end will not be affected, which is different when you use monolithic platform. This, in the long-term, can reduce your operation cost and makes your availability more stable.

Free to explore

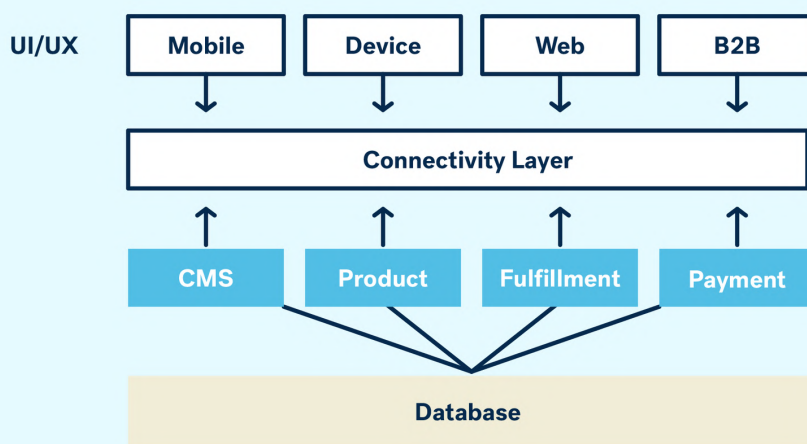
Your developers will be happy that they are no longer limited to certain language, framework or platform, they can use what they do best and let their creativity take over. As we see, unique experience requires your team to work out something different, if your designers, developers and marketers are allowed freedom to explore, and you will have less trouble to maintain the daily challenges of back-end, they will be able to learn new things and use new things much faster than before. Not to mention, you will have a faster time to market as well.

Fast and Secure

Most JAMstack e-commerce will have static sites that use CDN (content delivery network) to deliver faster to end users, no matter where they are. Smooth loading and interaction is the key for winning today's impatient users. JAMstack also greatly reduce the chance for malicious cyber attacks which can happen when your front-end and back-end are closely connected and your code or frameworks have some lethal weakness.

Easy to add new touchpoints

You want to add mobile apps? VR or AR shopping? Social shopping? What about IoT? How about Alexa? You name it. Since you now have API-based service, it's a small piece of cake to add something extra as your business grows and customer demands change. You do not need to create a whole new product just for that added experience, as they can consume data through the same API. You just focus on how to create the best UI and UX and run the top-notch marketing campaign to reach your sales goal.



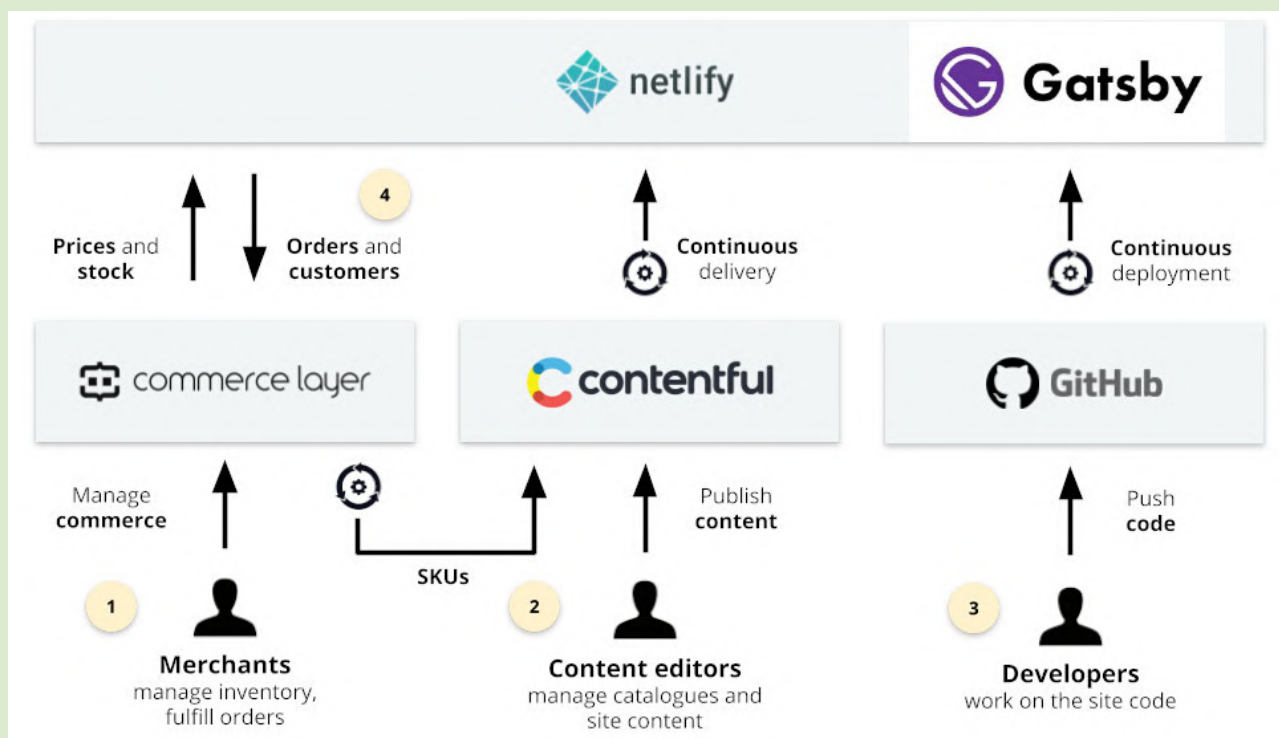
Petite and Minimal - a demo of JAMstack E-commerce Site

Petite and Minimal is a sustainable concept store that features minimal styles for petite ladies.

I've decided to use CommerceLayer as the commerce layer that connects database, manages merchandising, ordering and shipping etc.

Then I used Contentful as the CMS to enrich the data that's imported from CommerceLayer.

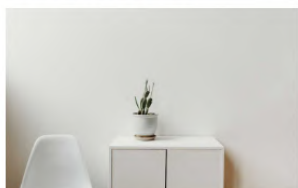
Finally I pulled data via GraphQL to my front-end which was generated by Gatsby, a static site generator so it will be super fast in the end. The code is in the Github and connected to Netlify for CI and CD.





CHOOSE STORE: [UK](#) [SWEDEN](#) [USA](#)

LATEST FROM OUR BLOG



MINIMAL HOME EDITION



KEEP WARM AND COMFY IN WINDY CITIES



HOW TO CARE FOR YOUR KNITWEAR

sign up to our newsletter

OK

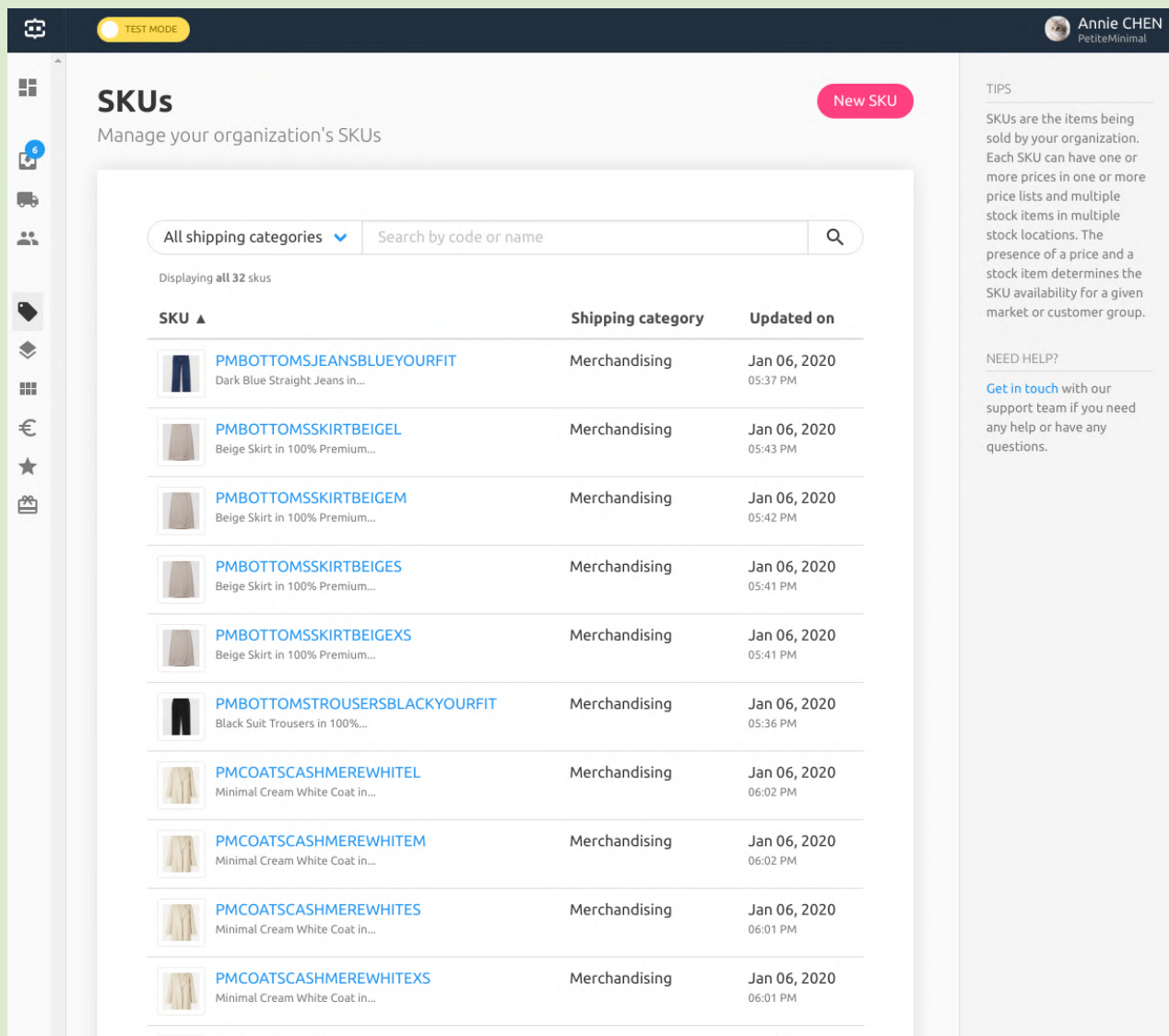
Follow us on social media



First, export data from database or other management tool such as Zapier to CommerceLayer. Then edit the data as needed. Here we need to have unique SKU(stock keeping unit)s, warehouses from different countries, quantities of products, their prices etc.

After everything is ready, we then export the data from here to Contentful CMS via commandline tool by setting up a yml file.

```
# .commercelayer-cli.yml
commercelayer:
  site: <your-base-endpoint>
  client_id: <your-client-id>
  client_secret: <your-client-secret>
contentful:
  space: <your-space-id>
  access_token: <your-content-management-access-token>
```



```
annie@TINA ~ $ commercelayer-cli export contentful
Warning: this will export your SKUs to Contentful. Continue? Y
Exporting SKUs to Contentful...
> PMBOTTOMSJEANSBLUEYOURFIT
> PMBOTTOMSSKIRTBEIGEM
> PMBOTTOMSSKIRTBEIGEL
> PMBOTTOMSSKIRTBEIGEXS
> PMBOTTOMSSKIRTBEIGES
> PMBOTTOMSTROUSERSBLACKYOURFIT
> PMCOATSCASHMEREWITEXS
> PMCOATSCASHMEREWITEL
> PMCOATSCASHMEREWHTES
```

Of course, before the last step, we need to have the data model ready in Contentful. We need to create country, catalog (per country), categories, product, product variant (that's different sizes or colors of one product), and sizes. After this is done, we then need to enrich the content such as edit the name, connect all the references, and also

write more detailed description. Contentful CMS allows Markdown rich text, so non-coder as content creator can work on an editor easily to write content.

So far the whole process is totally independent of our front-end, which is very different from the traditional method where everything is closely coupled.

Content model ? + Add content type

FILTER BY STATUS	Name	Description	Fields	Updated	By	Status
All	Catalogue		2	Sep 14, 2018	Me	ACTIVE
Changed	Category		4	Sep 20, 2018	Me	ACTIVE
Draft	Country		5	Sep 20, 2018	Me	ACTIVE
Active	Product		5	Sep 13, 2018	Me	ACTIVE
	Size		1	Sep 13, 2018	Me	ACTIVE
	Variant		5	Sep 13, 2018	Me	ACTIVE

PetiteMinimal ≡ Space home Content model Content Media Apps Settings ? 🔍 👤

Content ? **Product** Filter [Save as view](#) + Add entry ▼

[Scheduled Content](#)

Shared views My views

STATUS (4) ▼

- Published
- Changed
- Draft
- Archived

CONTENT TYPE (7) ▼

- Catalog
- Category
- Country
- Product**
- Size
- SKU
- Variant

11 entries found Usage: 185 / 5000 entries and assets

Name	Updated	Author	Status
Beige Skirt in Premium Silk	6 hours ago	Me	PUBLISHED
Black Suit Trousers in Wool	6 hours ago	Me	PUBLISHED
Minimal Cream White Coat in Cashmere	6 hours ago	Me	PUBLISHED
Light Beige Double Button Coat in Wool	6 hours ago	Me	PUBLISHED
Light Beige Casual Shirt Dress in Premium Silk	6 hours ago	Me	PUBLISHED
Dark Blue Straight Jeans in Organic Cotton	Thu, 7:45 PM	Me	PUBLISHED
Gray Blazer in Blend Linen and Pima Cotton	Thu, 7:43 PM	Me	PUBLISHED
White Blouse in Premium Silk	Thu, 7:43 PM	Me	PUBLISHED
White T-shirt with Organic Cotton	Thu, 7:43 PM	Me	PUBLISHED
Navy Blue Classic Sheath Dress	Thu, 7:41 PM	Me	PUBLISHED
Red Knit Dress in Blend Lyocell and Viscose	Thu, 7:41 PM	Me	PUBLISHED

Now we are all set to create the website. I used a Gatsby boilerplate I made from scratch to speed up the basic set up.

Then I first created the blog section. The blog has a posts folder where all the blog related pictures and markdown files are. They're generated via onCreateNode in the gatsby-node.js.

Then with onCreatePages, we pull data from GraphQL and programmatically create new nodes and new pages with the data that comes from our Contentful CMS.

GraphQL makes it easy to query data in a very flexible way. You only get data that's needed exactly for the page, which reduced extra size and also extra trips for server.

```
79     })
80   })
81   //----- above ends the blog page
82
83   //create products pages
84   const products = await graphql(`
85     query {
86       allContentfulCountry {
87         edges {
88           node {
89             code
90             node_locale
91             contentful_id
92             catalog {
93               name
94               node_locale
95               categories {
96                 name
97                 contentful_id
98                 products {
99                   contentful_id
100                  name
101                }
102                productsus {
103                  contentful_id
104                  name
105                }
106              }
107            }
108            marketId
109          }
110        }
111      }
112    }
113  `)
114
115
116  const edges =
117  products.data.allContentfulCountry.edges.filter(({ node }) => {
118    return node.node_locale !== -1
119  })
120
121  edges.forEach(({ node }) => {
122    const code = node.code.toLowerCase()
123    const locale = node.node_locale.toLowerCase()
124    if (locale === -1) return null
125    const catalogPath = env !== 'production' ? `/${code}/${locale}/` : `/${code}/${locale}/`
126    // console.log('catalog path', catalogPath)
127    //create catalog page - indicating the all products for a specific region/country
128    createPage({
129      path: catalogPath,
130      component: CatalogTemplate,
131      context: {
132        slug: catalogPath,
133        language: node.node_locale,
134        shipping: node.code,
135        pageTitle: node.code,
136        marketId: node.marketId
137      }
138    })
139
140    //create category page
141    node.catalog.categories.map(category => {
142      const categorySlug = category.name.trim().toLowerCase().replace(/\/s/gm, '-')
143      const categoryPath = env !== 'production'
144        ? `/${code}/${locale}/${categorySlug}`
145        : `/${code}/${locale}/${categorySlug}`
146      // console.log('category path: ', categoryPath)
147      // console.log('categoryId: ', category.contentful_id)
148      // console.log('categorySlug: ', categorySlug)
149      createPage({
150        path: categoryPath,
151        component: CategoryTemplate,
152        context: {
153          slug: categoryPath,
154          language: node.node_locale,
155          shipping: node.code,
156          categoryId: category.contentful_id,
157          categorySlug,
158          pageTitle: category.name.trim(),
159          marketId: node.marketId
160        }
161      })
162    })
163  })
164  })
165  })
166  })
167  })
168  })
169  })
170  })
171  })
172  })
173  })
174  })
175  })
176  })
177  })
178  })
179  })
180  })
181  })
182  })
183  })
184  })
185  })
186  })
187  })
188  })
189  })
190  })
191  })
192  })
193  })
194  })
195  })
196  })
197  })
198  })
199  })
200  })
201  })
202  })
203  })
204  })
205  })
206  })
207  })
208  })
209  })
210  })
211  })
212  })
213  })
214  })
215  })
216  })
217  })
218  })
219  })
220  })
221  })
222  })
223  })
224  })
225  })
226  })
227  })
228  })
229  })
230  })
231  })
232  })
233  })
234  })
235  })
236  })
237  })
238  })
239  })
240  })
241  })
242  })
243  })
244  })
245  })
246  })
247  })
248  })
249  })
250  })
251  })
252  })
253  })
254  })
255  })
256  })
257  })
258  })
259  })
260  })
261  })
262  })
263  })
264  })
265  })
266  })
267  })
268  })
269  })
270  })
271  })
272  })
273  })
274  })
275  })
276  })
277  })
278  })
279  })
280  })
281  })
282  })
283  })
284  })
285  })
286  })
287  })
288  })
289  })
290  })
291  })
292  })
293  })
294  })
295  })
296  })
297  })
298  })
299  })
300  })
301  })
302  })
303  })
304  })
305  })
306  })
307  })
308  })
309  })
310  })
311  })
312  })
313  })
314  })
315  })
316  })
317  })
318  })
319  })
320  })
321  })
322  })
323  })
324  })
325  })
326  })
327  })
328  })
329  })
330  })
331  })
332  })
333  })
334  })
335  })
336  })
337  })
338  })
339  })
340  })
341  })
342  })
343  })
344  })
345  })
346  })
347  })
348  })
349  })
350  })
351  })
352  })
353  })
354  })
355  })
356  })
357  })
358  })
359  })
360  })
361  })
362  })
363  })
364  })
365  })
366  })
367  })
368  })
369  })
370  })
371  })
372  })
373  })
374  })
375  })
376  })
377  })
378  })
379  })
380  })
381  })
382  })
383  })
384  })
385  })
386  })
387  })
388  })
389  })
390  })
391  })
392  })
393  })
394  })
395  })
396  })
397  })
398  })
399  })
400  })
401  })
402  })
403  })
404  })
405  })
406  })
407  })
408  })
409  })
410  })
411  })
412  })
413  })
414  })
415  })
416  })
417  })
418  })
419  })
420  })
421  })
422  })
423  })
424  })
425  })
426  })
427  })
428  })
429  })
430  })
431  })
432  })
433  })
434  })
435  })
436  })
437  })
438  })
439  })
440  })
441  })
442  })
443  })
444  })
445  })
446  })
447  })
448  })
449  })
450  })
451  })
452  })
453  })
454  })
455  })
456  })
457  })
458  })
459  })
460  })
461  })
462  })
463  })
464  })
465  })
466  })
467  })
468  })
469  })
470  })
471  })
472  })
473  })
474  })
475  })
476  })
477  })
478  })
479  })
480  })
481  })
482  })
483  })
484  })
485  })
486  })
487  })
488  })
489  })
490  })
491  })
492  })
493  })
494  })
495  })
496  })
497  })
498  })
499  })
500  })
501  })
502  })
503  })
504  })
505  })
506  })
507  })
508  })
509  })
510  })
511  })
512  })
513  })
514  })
515  })
516  })
517  })
518  })
519  })
520  })
521  })
522  })
523  })
524  })
525  })
526  })
527  })
528  })
529  })
530  })
531  })
532  })
533  })
534  })
535  })
536  })
537  })
538  })
539  })
540  })
541  })
542  })
543  })
544  })
545  })
546  })
547  })
548  })
549  })
550  })
551  })
552  })
553  })
554  })
555  })
556  })
557  })
558  })
559  })
560  })
561  })
562  })
563  })
564  })
565  })
566  })
567  })
568  })
569  })
570  })
571  })
572  })
573  })
574  })
575  })
576  })
577  })
578  })
579  })
580  })
581  })
582  })
583  })
584  })
585  })
586  })
587  })
588  })
589  })
590  })
591  })
592  })
593  })
594  })
595  })
596  })
597  })
598  })
599  })
600  })
601  })
602  })
603  })
604  })
605  })
606  })
607  })
608  })
609  })
610  })
611  })
612  })
613  })
614  })
615  })
616  })
617  })
618  })
619  })
620  })
621  })
622  })
623  })
624  })
625  })
626  })
627  })
628  })
629  })
630  })
631  })
632  })
633  })
634  })
635  })
636  })
637  })
638  })
639  })
640  })
641  })
642  })
643  })
644  })
645  })
646  })
647  })
648  })
649  })
650  })
651  })
652  })
653  })
654  })
655  })
656  })
657  })
658  })
659  })
660  })
661  })
662  })
663  })
664  })
665  })
666  })
667  })
668  })
669  })
670  })
671  })
672  })
673  })
674  })
675  })
676  })
677  })
678  })
679  })
680  })
681  })
682  })
683  })
684  })
685  })
686  })
687  })
688  })
689  })
690  })
691  })
692  })
693  })
694  })
695  })
696  })
697  })
698  })
699  })
700  })
701  })
702  })
703  })
704  })
705  })
706  })
707  })
708  })
709  })
710  })
711  })
712  })
713  })
714  })
715  })
716  })
717  })
718  })
719  })
720  })
721  })
722  })
723  })
724  })
725  })
726  })
727  })
728  })
729  })
730  })
731  })
732  })
733  })
734  })
735  })
736  })
737  })
738  })
739  })
740  })
741  })
742  })
743  })
744  })
745  })
746  })
747  })
748  })
749  })
750  })
751  })
752  })
753  })
754  })
755  })
756  })
757  })
758  })
759  })
760  })
761  })
762  })
763  })
764  })
765  })
766  })
767  })
768  })
769  })
770  })
771  })
772  })
773  })
774  })
775  })
776  })
777  })
778  })
779  })
780  })
781  })
782  })
783  })
784  })
785  })
786  })
787  })
788  })
789  })
790  })
791  })
792  })
793  })
794  })
795  })
796  })
797  })
798  })
799  })
800  })
801  })
802  })
803  })
804  })
805  })
806  })
807  })
808  })
809  })
810  })
811  })
812  })
813  })
814  })
815  })
816  })
817  })
818  })
819  })
820  })
821  })
822  })
823  })
824  })
825  })
826  })
827  })
828  })
829  })
830  })
831  })
832  })
833  })
834  })
835  })
836  })
837  })
838  })
839  })
840  })
841  })
842  })
843  })
844  })
845  })
846  })
847  })
848  })
849  })
850  })
851  })
852  })
853  })
854  })
855  })
856  })
857  })
858  })
859  })
860  })
861  })
862  })
863  })
864  })
865  })
866  })
867  })
868  })
869  })
870  })
871  })
872  })
873  })
874  })
875  })
876  })
877  })
878  })
879  })
880  })
881  })
882  })
883  })
884  })
885  })
886  })
887  })
888  })
889  })
890  })
891  })
892  })
893  })
894  })
895  })
896  })
897  })
898  })
899  })
900  })
901  })
902  })
903  })
904  })
905  })
906  })
907  })
908  })
909  })
910  })
911  })
912  })
913  })
914  })
915  })
916  })
917  })
918  })
919  })
920  })
921  })
922  })
923  })
924  })
925  })
926  })
927  })
928  })
929  })
930  })
931  })
932  })
933  })
934  })
935  })
936  })
937  })
938  })
939  })
940  })
941  })
942  })
943  })
944  })
945  })
946  })
947  })
948  })
949  })
950  })
951  })
952  })
953  })
954  })
955  })
956  })
957  })
958  })
959  })
960  })
961  })
962  })
963  })
964  })
965  })
966  })
967  })
968  })
969  })
970  })
971  })
972  })
973  })
974  })
975  })
976  })
977  })
978  })
979  })
980  })
981  })
982  })
983  })
984  })
985  })
986  })
987  })
988  })
989  })
990  })
991  })
992  })
993  })
994  })
995  })
996  })
997  })
998  })
999  })
1000  })
```

We create the front-end by writing different components and pulling different data from GraphQL.

Note the catalog, category, product, blog, and post pages are all programmatically generated. We only need to create one template for each. The rest can be distinguished by their unique slugs (part of the url).

Then we import the CommerceLayer component via a plugin and connect our front-end to the backend, so the price, availability and sizes will show up. We also use the ready made checkout system to speed up the time to market. Of course with more time and effort we can also build a customized version.

```
1 import React from 'react'
2 import * as CLayer from 'commercelayer-react'
3
4 import {usePriceLoading} from '../hooks'
5
6 import loader from '../images/loader.svg'
7
8 const Product = props => {
9   const { data, onClick} = props
10
11   const loading = usePriceLoading('clayer-prices-ready')
12
13   const srcImg = `https://${data.image.file.url}`
14   const variants = data.variants.map(variant => {
15     return {
16       code: variant.code,
17       name: variant.name,
18       label: variant.size.name
19     }
20   })
21   // console.log(variants)
22
23   const amountProps = {
24     amount: {
25       className: 'has-text-success'
26     },
27     compare: {
28       className: 'has-text-grey-light'
29     }
30   }
31
32   return (
33     <div className="product-grid">
34       <img className="product-img" src={srcImg} alt={data.name} />
35       <div className="product_info">
36         <h1 className="product_info_title">{data.name}</h1>
37         <div className="product_info_desc"
38           dangerouslySetInnerHTML={{ __html: data.description.childMarkdownRemark.html}}></div>
39
40         { /* price */ }
41         <div className="product_info_price">
42           <CLayer.Price
43             skuCode={data.variants[0].code}
44             AmountProps={amountProps}
45           />
46         {loading ? <img src={loader} alt="loader" width="50" /> : null}
47       </div>
48
49       { /* variant parameters such as size or color */ }
50       <div className="product_info_sizes">
51         <CLayer.VariantSelect
52           PriceContainerId="price"
53           AvailabilityMessageContainerId="availability-message"
54           AddToBagId="add-to-bag"
55           promptText="select size"
56           skus={variants}
57         />
58       </div>
59
60       <CLayer.AvailabilityMessageContainer id="availability-message" />
61
62       <CLayer.AvailabilityMessageAvailableTemplate
63         className="available-message has-text-success"
64         availableTemplate={
65           <p className="has-text-success">
66             <span>Available</span>{ ' ' }
67             in{ ' ' }
68             <CLayer.AvailabilityMessageMinDays className="available-message-min-days" />
69             -
70             <CLayer.AvailabilityMessageMaxDays className="available-message-max-days" />{ ' ' }
71             days
72           </p>
73         }
74       />
75
76       <CLayer.AvailabilityMessageUnavailableTemplate
77         className="unavailable-message has-text-danger"
78         unavailableTemplate={<p>Unavailable</p>}
79       />
80
81       <CLayer.AddToBag
82         className="button"
83         id="add-to-bag"
84         AvailabilityMessageContainerId="availability-message"
85         text="BUY"
86       />
87     </div>
88   )
89 }
```



BEIGE SKIRT IN PREMIUM SILK



BLACK SUIT TROUSERS IN WOOL



MINIMAL CREAM WHITE COAT IN CASHMERE



LIGHT BEIGE COAT IN



LIGHT BEIGE CASUAL SHIRT DRESS IN PREMIUM SILK



DARK BLUE STRAIGHT JEANS IN ORGANIC COTTON



GRAY BLAZER IN BLEND LINEN AND PIMA COTTON



WHITE SHIRT



WHITE T-SHIRT WITH ORGANIC COTTON



NAVY BLUE CLASSIC SHEATH DRESS



RED KNIT DRESS IN BLEND LYOCCELL AND VISCOSE

acornblaze
€2,035.00

Light Beige Double Button Coat In 100% Wool M €2,035.00 [remove](#)

[Continue Shopping](#)

[Proceed to checkout](#)

Follow us on social media



LIGHT BEIGE DOUBLE BUTTON COAT IN WOOL

- Lorem ipsum dolor sit amet, consectetur adipiscing elit.
- Aenean vel quam quis velit tincidunt faucibus semper vel risus.
- Nullam lobortis arcu non fringilla sodales.
- In hendrerit massa vel purus gravida cursus.
- Duis dignissim leo quis ullamcorper lacinia.

Ut non purus in tellus rutrum aliquet eu et ex. In egestas velit vel condimentum luctus. Nulla pellentesque nisi vel sem commodo hendrerit sit amet in leo. Suspendisse dignibus lorem vel libero dignissim finibus. Morbi id velit ac urna malesuada mollis eget id diam. Curabitur et sapien a risus suscipit malesuada.

€2,035.00

select size

BUY

Follow us on social media





Thank you, Sheldon!
Your order is confirmed

Customer:
sheldon@cooper.com

Billing address:
Sheldon Cooper
1 Stockholm Street
112 76 Stockholm (Stockholm) Sweden
54962937

Shipping address:
Sheldon Cooper
1 Stockholm Street
112 76 Stockholm (Stockholm) Sweden
54962937

Shipment 1 of 1



Light Beige Double Button Coat in 100% Wool M
PMCOATSWOOLBEIGEM
Quantity: 1

Standard Shipping

[CONTINUE SHOPPING](#)

[Terms of service](#) - [Privacy policy](#)

Order summary

€2.035,00

Order number: #163777



Light Beige Double Button Coat in 100% Wool M
PMCOATSWOOLBEIGEM
Quantity: 1

Subtotal	€2.035,00
Discount	€0,00
Shipping	€0,00
Payment method	€0,00
Taxes	€0,00
Gift card	€0,00

Order total €2.035,00

Dashboard

ORDER FULFILLMENT

Orders

Shipments

Customers

PRODUCT OFFERING

SKUs

SKU options

Inventory

Prices

Promotions

Gift cards

Settings

All orders

Order #163777

Placed on Jan 16, 2020

[Cancel order](#)

[Approve order](#)

STATUS

Order, payment, and fulfillment status.

placed Jan 16, 2020 1 transaction

authorized 1 transaction

unfulfilled 1 shipment

BASIC INFO

Market, shipping country and customer preferred language.

Country Sverige

Market Europe

ORDER SUMMARY

This order contains 1 line item.

Light Beige Double Button Coat in 100% Wool M
PMCOATSWOOLBEIGEM X1 €2.035,00

Subtotal €2.035,00

1 line item

Discount €0,00

No promotions applied

Adjustments €0,00

Not applied

Shipping method(s) €0,00

Standard shipping

Payment method €0,00

Wire transfer

Taxes €0,00

Not calculated

Gift card €0,00

Not applied

Order total €2.035,00

CUSTOMER

Email address and customer group

sheldon@cooper.com

Language: English

SHIPPING ADDRESS

The customer shipping address is used in all the order shipments.

Sheldon Cooper
1 Stockholm Street - 112 76 Stockholm (Stockholm) - SE
Phone: 54962937

BILLING ADDRESS

The customer billing address can be imported into any 3rd party invoicing system.

Sheldon Cooper
1 Stockholm Street - 112 76 Stockholm (Stockholm) - SE
Phone: 54962937

PAYMENT SOURCE

Used for all the order's transactions

wire_transfer #1272

WT

TRANSACTIONS

All the order's authorizations, captures, voids, or refunds.

Authorization succeeded Jan 16, 2020

Not applied €2.035,00 11:07 AM

SHIPMENTS

Orders can be split into many shipments, depending on the inventory model.

#163777/5/001 upcoming Jan 16, 2020

0 parcels 11:07 AM

INTEGRATION

Use the reference fields to attach any external identifier to this resource, so that it can be easily filtered through the API.

Reference

Optional
Any external identifier that might be useful to link this resource to other systems through the API.

[Update reference](#)

METADATA

Use metadata to attach additional information to this resource, either manually or through the API.

This order metadata is empty. Populate the metadata through the API to get them available for update in this form.

ATTACHMENTS

Use attachment to link external files to this resource through the API.

This order has no attachments. Create some attachments through the API to get them visible in this form.

RESOURCE ID

q8AV9mWdW

RESOURCE TYPE

orders

REFERENCE

163777

CREATED

Jan 16, 2020 11:06 AM

UPDATED

Jan 16, 2020 11:06 AM

PLACED

Jan 16, 2020 11:07 AM

APPROVED

11:07 AM

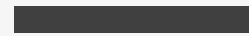
CANCELLED

11:07 AM



Conclusion

JAMSTACK IS THE FUTURE OF E-COMMERCE TO CREATE FAST AND UNIQUE DIGITAL STORE.



Overall, I think JAMstack is worth exploring. The downside is that it might require creative skills from the front-end developer to design beautiful layout and they need to be familiar with micro-services and APIs. The service fee might also be more expensive from start for some small business. But for business that does plan to scale and expand in the future, especially they might need different countries, currencies, languages, this stack does allow flexibility, speed and excellent developer experiences, which will result in a great product every customer loves to use.



References



1. *JAMstack site*. <https://jamstack.org/>, Nov, 2019.
2. *WTF is JAMstack?* <https://jamstack.wtf/>, Nov, 2019.
3. Mathias Biilmann, Phil Hawksworth. *Modern Web Development on the JAMstack*. United States of America: O'Reilly. June, 2019.
4. Susan Moore. *Top 10 Trends in Digital Commerce* <https://www.gartner.com/smarterwithgartner/top-10-trends-in-digital-commerce/>. Gartner, Oct 12, 2019.
5. Kelly Goetsch. *Microservices for Modern Commerce*. United States of America: O'Reilly. 2017.

