

React & Next.js Meetup by Devstaff Community

Integrate Next.js React Framework & Contentful CMS via GraphQL API

Presented by Front-end Chapter, Vodafone Greece

13 March 2023

Who we are

- Ioanna Parga – Front-end Developer
- Giannis Vasilopoulos – Chapter Lead Dev. Front-end
- Dimitris Matsis – Front-end Developer

HELLO!
HELLO!
HELLO!



Agenda

01



Agenda

- The Vodafone team & Vodafone Tech Hub in Crete
- Next.js - The React Framework
- Contentful – As a next-gen Headless CMS
- GraphQL – A modern query language for your API



The Vodafone Tech team & Tech Hub in Crete

02



The Vodafone Tech team

- A diverse team of 300 people
- Agile way of working
- Various Chapters based on specialty
 - Android, IOS, DevOps, Back-ends, SAs and much more
- Front-end Chapter
 - **29** members
 - **5** of them working on the Tech Hub in Crete
 - Keep growing



Vodafone Tech Hub

Heraklion, Crete

- Counting: 18 members (and keep growing)
- Various levels of seniority (e.g. Tech Leads, Chapter Leads, Developers, Graduates, Interns)
- Hybrid working model (60% home – 40% office)



Next.js - The React Framework

03



Next.js - The React Framework

- Full-stack React applications made easy
- How?
 - Rich User Experience (easier and faster)
 - Great performance (also easier and faster)
 - Rapid feature development
 - Rich documentation
 - Great community support

NEXT.js



Next.js - The React Framework

- Full-stack React applications made easy
- Why?
 - Zero Config
 - Development & Production Build System
 - Automatic Code Splitting
 - Pre-render through Server-Side Rendering
 - Static Site Generation (Incremental Static Regeneration)
 - Routing
 - API Routes
 - Fast Refresh (HMR)
 - Typescript support

NEXT.js



Next.js - Client-Side Rendering (CSR)

- HTML of the page is generated on the client
- Rendering work happens on the user's device
- No SEO friendly

NEXT.js



Next.js - Server-Side Rendering (SSR)

- HTML of the page is generated on server for **each** request
- JSON data and JavaScript make components interactive on the client
- SEO friendly

NEXT.js



Next.js - Static Site Generation (SSG)

- HTML of the page is generated on the server, **once on build-time**
- JSON data and JavaScript make components interactive on the client
- Create or update static pages through **Incremental Static Regeneration (ISR)**
- On-demand Revalidation
- No need to rebuild your entire site if your data changes
- SEO friendly

NEXT.js



Next.js - Routing

- File-system based router

- /pages/index.tsx
- /pages/about.tsx

- Dynamic routes

- /pages/blog/[slug].tsx

- Client-side navigation

- next/link
- next/router

```
import Link from 'next/link'

function Home() {
  return (
    <ul>
      <li>
        <Link href="/">Home</Link>
      </li>
      <li>
        <Link href="/about">About Us</Link>
      </li>
      <li>
        <Link href="/blog/hello-world">Blog Post</Link>
      </li>
    </ul>
  )
}

export default Home
```



Next.js - Internationalized Routing i18n

- Automatic locale detection
- Serving language based on sub-path or domain

```
    domain: 'example.fr',
    defaultLocale: 'fr',
  },
  {
    domain: 'example.nl',
    defaultLocale: 'nl-NL',
    // specify other locales that should be redirected
    // to this domain
    locales: ['nl-BE'],
  },
],
},
}
```

For example if you have `pages/blog.js` the following urls will be available:

- `example.com/blog`
- `www.example.com/blog`
- `example.fr/blog`
- `example.nl/blog`
- `example.nl/nl-BE/blog`

Sub-path Routing

Sub-path Routing puts the locale in the url path.

```
// next.config.js
module.exports = {
  i18n: {
    locales: ['en-US', 'fr', 'nl-NL'],
    defaultLocale: 'en-US',
  },
}
```

With the above configuration `en-US`, `fr`, and `nl-NL` will be available to be routed to, and `en-US` is the default locale. If you have a `pages/blog.js` the following urls would be available:

- `/blog`
- `/fr/blog`
- `/nl-nl/blog`

The default locale does not have a prefix.



Next.js - Built-in components

- next/font
 - Optimize fonts without layout shift
- next/script
 - Load third-party script anywhere
- next/dynamic
 - Lazy load support for components

NEXT.js



Next.js - Image component

- Minimize payload
- Reduce layout shift
- Load when in viewport
- Blurred placeholder
- Compression

```
import Image from 'next/image'

export default function Home() {
  return (
    <div>
      <h1>My Homepage</h1>
      <Image
        src="/me.png"
        alt="Picture of the author"
        width={500}
        height={500}
      />
      <p>Welcome to my homepage!</p>
    </div>
  )
}
```



Next.js - Preview mode

- Preview draft content
 - API Route
- Extensive CMS support

NEXT.js



Next.js = React ++

- Faster webpages for the user
- Quicker development time (HMR)
- CSR/SSR/SSG
- Built-in CSS support
- Better scalability
- One unified codebase
- React/Typescript compatibility

NEXT.js



Contentful – As a Headless CMS

04



Contentful - Goodbye "monolithic" CMS

- What is Contentful?
 - Next-gen Headless CMS
 - SaaS
 - Administration panel to create and manage content
 - Supports Content versioning
 - Supports Content scheduling
 - Role-based user permissions
 - Fully customizable content structure and management UI



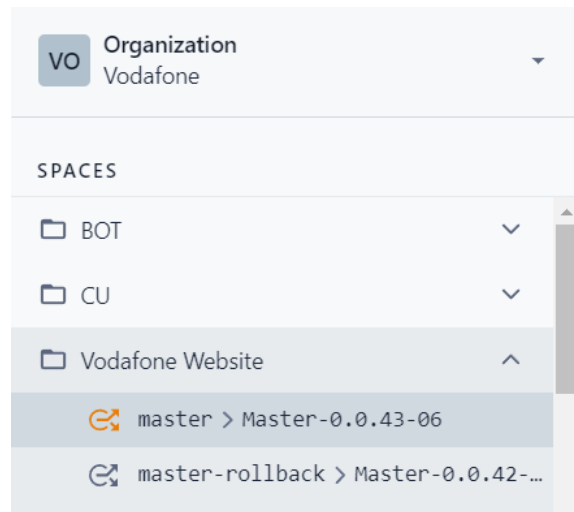
Contentful – Differences from common CMS

- Content infrastructure
 - CDN takes care of delivering content to end-users 24/7 (out-of-the-box)
- Manage flexible content models
 - There are no pre-defined content models. You create the content model yourself
- Uncluttered UI
 - Offers you a beautiful and simple user interface, achieving best UX
- Mobile and web, all with the same system



Contentful – Organization / Space / Environment

- Organization
 - The administration of spaces
- Space
 - A workspace that contains all content and media for a project
- Environments
 - Entities within a space



Contentful – Content model & Content

- Content model
 - It is the structure of our content and semantic relationships between key content types.
- Content
 - It is the actual information for our model structure
- Steps to deliver content from Contentful
 - Define your **Content Model**
 - Create **Content**
 - Deliver your content via API

Fields (6)						Groups	JSON preview	Sidebar	Entry editors
⋮	Ab	Internal name	Short text		Entry title	Settings	...		
⋮	Ab	Slider text	Short text			Settings	...		
⋮	Image	Image	Media			Settings	...		
⋮	Text	Text	Rich text			Settings	...		
⋮	Button	Button	Reference			Settings	...		
⋮	Countdown Timer	Countdown Timer	Reference			Settings	...		



Contentful - Goodbye "monolithic CMS"







- Why Contentful?

- Cloud based CMS solution
- User-friendly, fast and robust interface
- Channel agnostic (same content across platforms/channels - Web, MVA)
- GraphQL Ready
- Multiple coding languages SDK (easy integration with React, NextJS, NodeJS and next-gen frameworks)
- Simple/Flexible Content modeling (in a few clicks)
- Multiple Locales
- Media support and advanced image API
- Content As Code (CLI)
- Robust Environment creation
- Quick and Efficient Rollback process
- Supports Webhooks



Contentful - Custom Apps

- Contentful provides an **App Framework** to serve development needs and optimize our editorial experience
- What are apps?
 - Third party application, enhance the functionality and usability of Contentful, allow users to customize their workflows and optimize their content
- Common use cases
 - Custom editors
 - Integration with external services(ex eCommerce)
 - Workflow automation
 - Analytics and reporting
 - Personalization

	SEO Tools ✓	CUSTOM	...
	broadleaf ✓	CUSTOM	...
	color-picker ✓	CUSTOM	...
	media-svg ✓	CUSTOM	...
	page-url ✓	CUSTOM	...
	repeater ✓	CUSTOM	...



Contentful – Vodafone Key-Value pairs Custom App

- Key-Value pairs

Requirement:

- Contentful doesn't provide a field type that returns a key value pair, resulting to increased query complexity and slow query speed.
- Need to convert the query results to key value pairs inside the app resulting to unnecessary processing.

Solution:

- It creates a repeatable list of key-value pairs in the Contentful web app. Actually, it is a Front-end app that utilizes the default json field and extends Contentful's default interface.

The screenshot displays the Contentful web app interface for a field named 'TranslationsObj'. The field is of type 'Text' and is currently empty. A hint text 'Valid html tags strong, em, span, a, u, br' is visible. Below the field, there is a '+ Add Item' button. A 'Click to Expand/Collapse' link is also present. The field is currently expanded, showing a 'Key' field with the value 'forgotUsername' and a 'Value EL' field with the value 'Ξέχασα το όνομα χρήστη'.



Contentful – Vodafone Page URL Custom App

- Page URL

Requirement:

- Need for parent-child relationship for the pages to cover our needs for breadcrumbs and automatic url creation

Solution:

- It autocompletes an URL field with its parent reference URL if any. If a parent URL change, then all child pages URL should be updated

Ab Url Symbol Settings Validation Default value Appearance X

Choose how this field should be displayed

	Slug	broadleaf	color-picker	page-url	UI EXTENSION
selected	generated slug	gear icon	gear icon	gear icon	gear icon
label	Slug	broadleaf	color-picker	page-url	Select

Cancel Confirm

Internal name (required)
TOBi Login
10 characters Maximum 256 characters

Slug (required)
tobi/login

Url
/tobi/login



Contentful – Vodafone Color-picker Custom App

- Color Picker

Requirement:

- Contentful doesn't provide a friendly way to handle a field for hex colors and to strict the author to choose specific colors.

Solution:

- It is an implementation to enhance our editorial experience and use a color picker for fields that we need to fill with Hex Colors.

The screenshot shows a user interface for a color picker. At the top, there is a label "Content Text Color" above a text input field containing the hex code "#e60000". To the right of the input field is a small red square color swatch. Below the input field is a horizontal row of 18 circular color swatches. The first row contains 17 swatches: red, white, dark gray, teal, dark teal, light blue, purple, dark purple, olive green, yellow, orange, dark red, green, and black. The second row contains 1 swatch: dark gray. Below the color swatches is a label "Background Image" followed by a dashed line and a small rectangular placeholder box.



Contentful – Vodafone SEO Tools Custom App

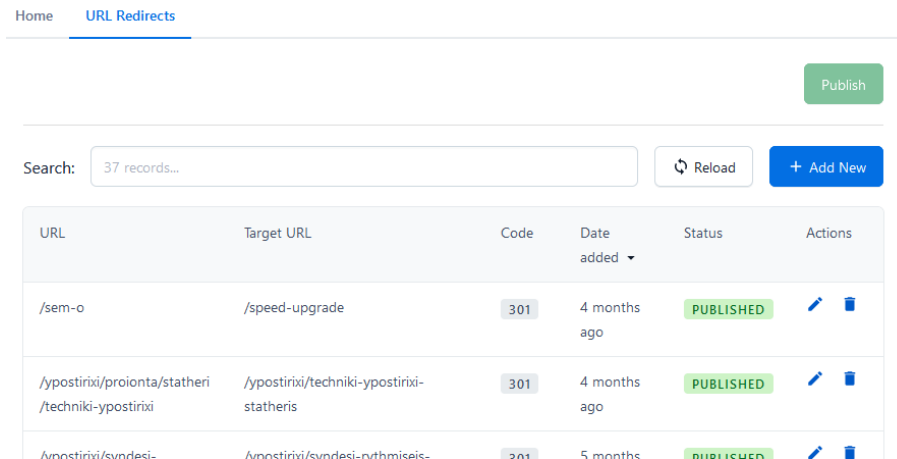
- SEO Tools

Requirement:

- Martech team needed to have a tool for global SEO needs such as maintaining a list of URL redirects for SEO reasons, handle robots.txt doc etc.

Solution:

- We have built the URLs redirection list component. Each redirect is a Contentful entry and when the author publishes the list a webhook is triggered. The webhook calls a λ function in our AWS environment. The λ function updates the list of URL redirects in our load balancer.



Home URL Redirects					
<div>Publish</div>					
<div>Search: 37 records... Reload + Add New</div>					
URL	Target URL	Code	Date added	Status	Actions
/sem-o	/speed-upgrade	301	4 months ago	PUBLISHED	
/ypostirixi/proionta/statheri /techniki-ypostirixi	/ypostirixi/techniki-ypostirixi-statheris	301	4 months ago	PUBLISHED	
/annetirixi/sundesi-	/annetirixi/sundesi-nuthmicic-	301	5 months	PUBLISHED	



GraphQL

05



GraphQL

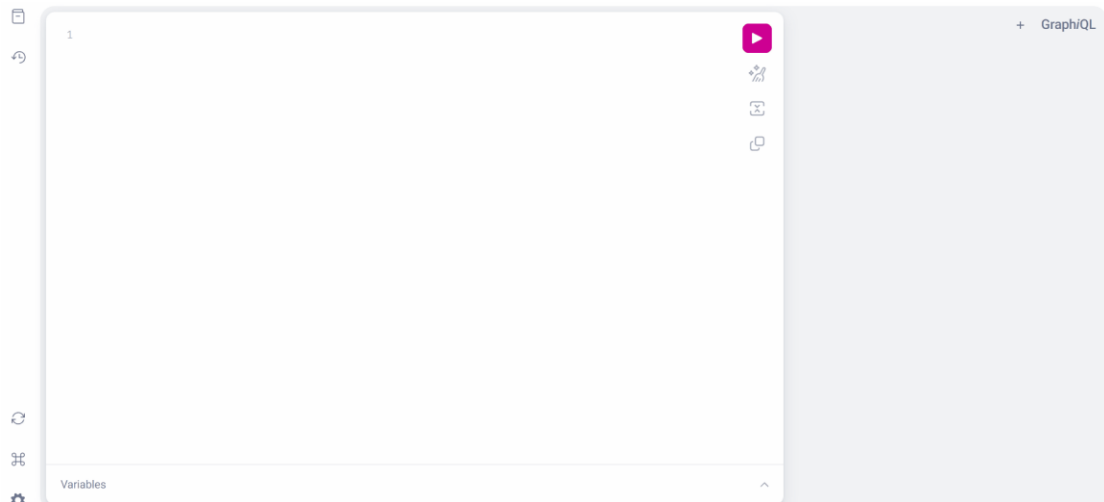
What is GraphQL?

- A modern and mature alternative to REST APIs
- Open source. Developed by Facebook
- **Why GraphQL?**
 - Efficient data retrieval
 - Flexible queries
 - Strong typing
 - Improved developer experience



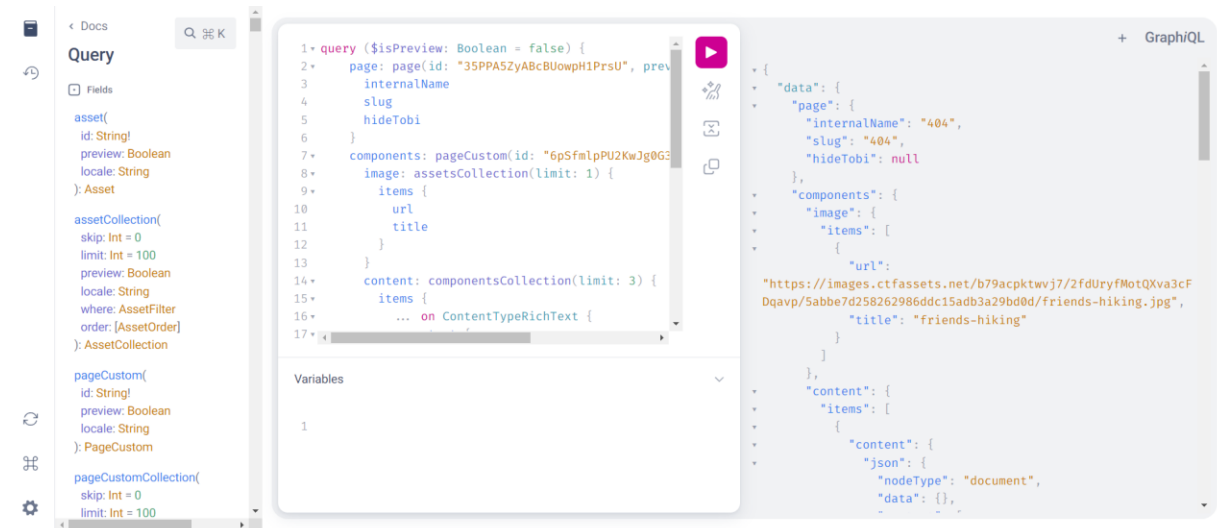
Setting up GraphQL API with Contentful

1. Add content models and content to our space
2. Use GraphQL to :
 - Query Contentful, (browser / Contentful custom app)
 - Test our queries and ensure that they return the expected data
3. Integrate our GraphQL API in next.js project, using GraphQL client to handle GraphQL queries
4. GraphQL response error handling

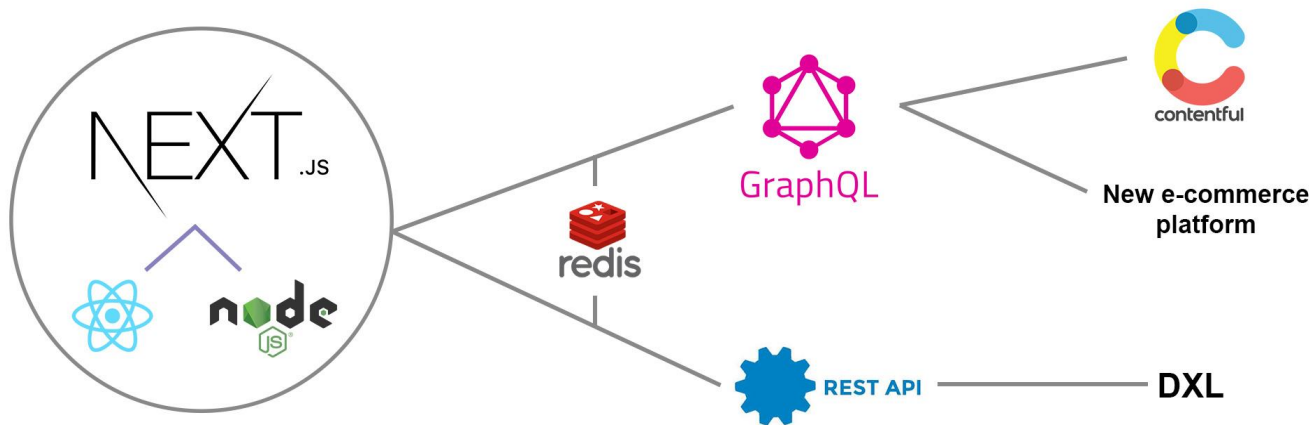


Practices

1. Query complexity limits
2. Filtering
3. Fragments(reusability purpose)
4. Use caching mechanism(server,client)



Open-source / Reusability / Scalability



Next.js (React)

- Rendering

Next.js (Node)

- Core web functionalities
- Data aggregation
- Routing
- Business logic
- APIs
- Authentication
- Session management

Redis

- Caching
- Session Storage

Contentful

- Marketing data / Translations
- Authoring Interface (CMS)



Together we can