University of Hradec Kralove
Faculty of Informatics and Management
Department of Information Technology

Alternatives to the content management system for a consulting firm

Bachelor’s paper

Autor: Denys Danylko

Studijní obor: Informational Management

Vedoucí práce: Mgr. Daniela Ponce, Ph.D.

Hradec Kralove April 2022

Statement:
I declare that I have prepared my bachelor thesis independently and using the literature mentioned below.

In Hradec Králové, 27 January 2022 Danylko Denys

Annotation:

The topic of the bachelor's work is comparing two different architecture solutions: JAMStack and traditional Content Management System - CMS.

The main objective of the work is to show how progressive new technologies are and why we should learn them and why we should create websites on JavaScript frameworks, but not on PHP.

The work has the following main parts: introduction to WordPress, WordPress advantages and disadvantages, methodology procedure, alternatives, JAM Stack as the best alternative (will be added later), transferring consulting site from CMS to JAM Stack, suggestions and recommendations for reader.

The main contribution of the work is to help users of WordPress to recognize that there are existing more effective and more convenient alternatives for creating websites. This way, alternative websites will be faster and safety.

Table of Contents

1. Introduction

JAMstack is a modern web development architecture. It is not a programming language or any form of tool. It is more of a web development practice aimed toward enforcing better performance, higher security, lower cost of scaling, and better developer experience.

Often developers, business owners, governments, and other people get faced with different errors on web-site, which are based on CMS, for example, WordPress. Between "content" and "database" is a big gap, and due to which sometimes crashes occur and the site stops working properly. To solve such problems, humanity created JAM (JavaScript, API, Markup). Architecture, that will increase the speed of your site and get rid of errors.

JAMStack is modern architecture, which is based on client-side JavaScript, reusable APIs, and prebuilt Markup. Actually, when we talk about "The Stack", we no longer talk about technologies, operating systems, some web servers, or databases. The JAMStack is not about specific technologies. It is a new approach to thinking, building websites and apps that delivers better performance, higher security, lower cost of scaling, and a better developer experience. - according to JAMStack documentation [10]

JAMStack is a new trend in the web development sphere that was created by Mathias Biilman, the CEO and co-founder of Netlify (providing hosting services).

However, stacks generally are just a combination of several technologies used to create a web or mobile application. So JAMstack is the combination of JavaScript, APIs, and Markup.

This work was created to prove and describe new technologies such as JAMStack architecture compared to architecture solutions of CMS.

2. The aim of the bachelor's work

There are a few aims of this bachelor's work. First of all, to compare and evaluate JAMStack architecture and architecture solutions of traditional CMS. I will compare by evaluation criteria, which will be described in further parts.

Secondly, to compare the way of rendering pages and architectures at all. There are 2 different types of rendering pages - dynamic and static.

The next point will be deciding which approach will be better in the following situations: a small company that is just getting started, a fast-growing startup or a B2B business, and a bigger company able to hire developers.

Thirdly, describe the advantages and disadvantages of JAMStack and traditional CMS. How it performs, speed, costs for development, analysis of approaches, and the way of interactions with other tools. Also, compare environments. Finally, decide which option will be better for the end-user and why.

Finally, to create a practical project for this bachelor's work, where will be implemented all necessary functions for the reason to have some statistics from both architectures. This project will be described in further parts too.

3. Processing methodology

The main objective of this research paper is to analyze comparative architecture solutions - JAMStack and traditional CMS. This type of research method was selected for a few reasons, as the application of secondary data furnishes benefits, and several studies are done on such a topic before. The purpose of choosing this method was to acquire a broader understanding of the topic. By utilizing secondary data as the source for the study, the scope of the data was greater, than it would have resulted with the use of only the primary data.

The way of studying this topic is the way analyzing next important criterias:

- advantages and disadvantages of approaches
- situations, where architectures can be applied
- speed, performance, cost, time-consuming, SEO-extension and others
- creating own project based on requirements for investigating the problem
- researching for theory parts to make a better understanding

The conclusion will be drawn from analyzing various secondary data resources. They were used in order to draw the basis for following development of the paper theory. Resources was used to build basement for further research. For selecting such recources author of the work was using external sources. Author of work was searching relevant articles on his topic from the most trusted sources. Often, it was blogs, research articles, that is why the topic is innovative. Paper author was interested in topic of this research , because hi is working in this sphere and as a result, it helps research to be more useful.

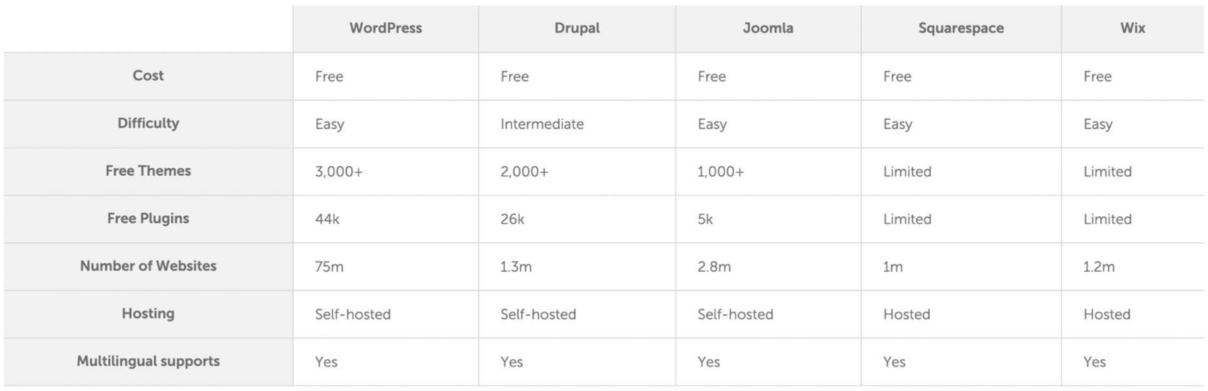
4. Traditional CMS

CMS - Content Managemant System, in most cases is software that helps users create, manage, and modify content on a website without the need for specialized technical knowledge.

4.1. What is CMS about?

Architecture CMS solutions powers both the **backend** of the website (the interface where a user logs in to make changes or add new content) and the **frontend** (the visible part of the website that your visitors see on the web).

CMS platform is the most popular because it's the easiest for non-developers to use to build a professional-looking website. [4]



	WordPress	Drupal	Joomla	Squarespace	Wix
Cost	Free	Free	Free	Free	Free
Difficulty	Easy	Intermediate	Easy	Easy	Easy
Free Themes	3,000+	2,000+	1,000+	Limited	Limited
Free Plugins	44k	26k	5k	Limited	Limited
Number of Websites	75m	1.3m	2.8m	1m	1.2m
Hosting	Self-hosted	Self-hosted	Self-hosted	Hosted	Hosted
Multilingual supports	Yes	Yes	Yes	Yes	Yes

Figure 1: Comparison of WordPress and other CMS [4]

There are a lot of CMS platrforms exisitng, as it mentioned on the picture below. Let's take an example, the most popular CMS nowadays is WordPress. It's quicker to master than most CMS; without much effort, anyone can build a powerful website that's above all easy to manage. WordPress themes and plugins can add new design options and added functionality. Considering the information above, the user can create any website he wants. And it does not matter which skills this user has for a moment. Unfortunately, all the best things in the world have bad qualities. So, WordPress has such advantages:

- **Simplicity** – WordPress allows you to publish and build your website content quickly.
- **The cost is free** – Users only need to pay for web hosting and a domain name.
- **Flexibility** – WordPress allows you to create many types of websites from personal blogs and online stores to online magazines and newspapers.
- **Easy to use** – If you can use a word processing software like Microsoft Word, you can use WordPress to build and manage a website.
- **Extendable with WordPress plugins** – The core WordPress software can be extended with WordPress plugins. WordPress plugins are bits of software that you can upload to your website to add more features (such as e-commerce, SEO, backups, contact forms, and more). There are thousands of free WordPress plugins available on the WordPress.org plugin directory and a thriving premium (paid) plugins industry.

- **Highly customizable with WordPress themes** – WordPress themes provide the design and layout of your website. With a click of a button, you can change the entire look of your website by applying a new WordPress theme. More advanced WordPress themes are more like WordPress page builders, giving you even more control of your layouts.

And has such disadvantages:

- **Based on PHP** – comparing the syntax of PHP and JavaScript we can make a decision, that second more modern and progressive. PHP is an old programming language and the need for it rapidly decreasing because of new alternatives.
- **Load loading** – WordPress is loading the exact page by the way of loading all pages, while JAM Stack websites are loading one exact page. This way, we increase the speed of the website.
- **Not easy to use** – at the same time, WordPress is a complicated system, where is where there are a lot of extra functions, that simple user will not use for everyday work.

5. JAM Stack

JAMstack is a software architecture and philosophy that adheres to the following components: **Javascript**, **APIs**, and **Markup**.
In other words, serving HTML from static hosting or CDN (content delivery network), but providing dynamic content and an interactive experience through JavaScript.

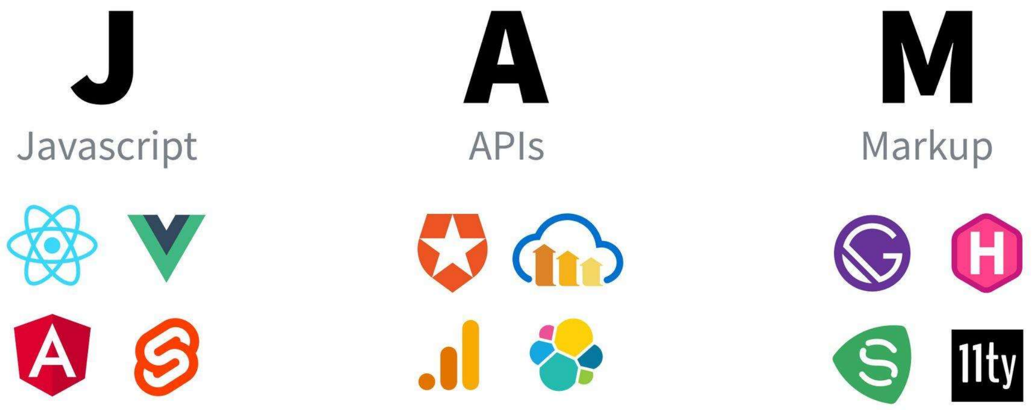


Figure 2: JAMStack [8]

This concept started from an idea: that the developer community needed a way to get past the potential negative stigmas of the word “static”. It gives us a term to describe how a modern dynamic web app is built.

Some argue we’re past the JAM in Jamstack, but the concepts remain. We still depend on Markup to deliver a website or application and optionally can use JavaScript and APIs to enhance the experience. [8]

So, in other words, JAMStack is an idealogy, which was created for replacing old CMS systems, and instead of the word “static”, now we have “dynamic”, which is more convenient for end-users and for developers. So, JAM is JavaScript, API, and Markup. And Stack is about architecture, which opens a more precise look at dynamic web apps. So, let’s go deeper in detail.

The JAMstack architecture consists of four elements:

- Site Generators
- Headless CMS,
- Arontend framework,
- CDN

The generators are in charge of creating static pages, which are sent to the user's browser via CDN. Because JAMstack is independent, you can install a variety of technologies, such as Next.js, Hugo, Gatsby, Jekyll or Nuxt. It is similar in the case of CMS, where you can reach for Strapi, Netlify or Webina, for example.

Frontend frameworks are closely related to JavaScript from the name of the architecture. Addresses dynamic site functionality, on-page scripting, and API communication. They figure on the client side, where the dynamic content is rendered. This makes websites and web applications built on the JAMstack architecture faster and therefore more user-friendly. Examples of front-end frameworks include Vue JS, the React library, or Angular.

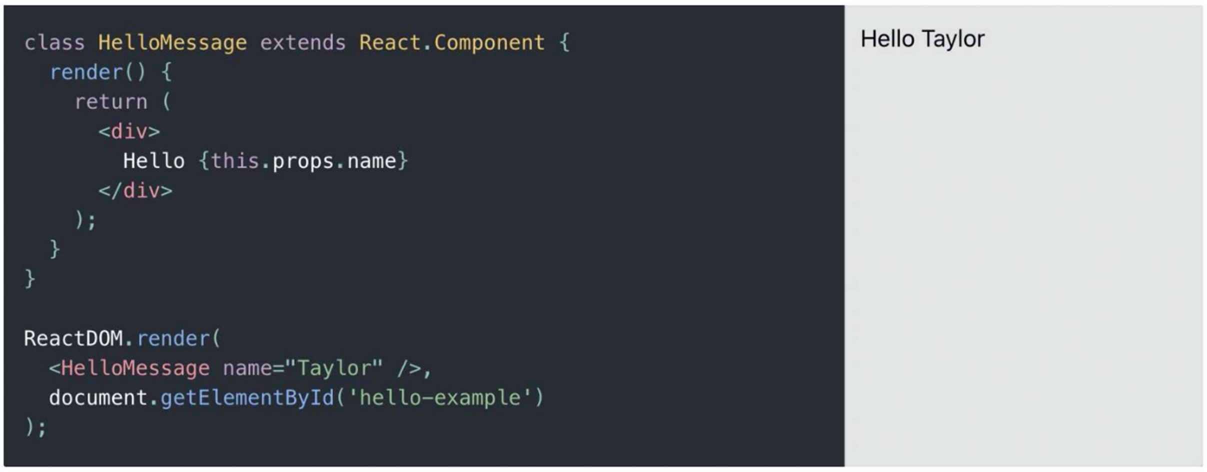
The API or application interface is in charge of communication between the individual components and serves as a replacement for the backend part of the application. For example, they handle forms or security issues. Each developer has the opportunity to write their own interface or use ready-made third-party solutions.

The last part of the architecture is the Markup language. It is a content part, consisting of HTML, CSS and JavaScript, which is basically a presentation layer of a website or application.

What is JAM?

JavaScript

The component that's probably done the most work to popularize the JAMstack is Javascript. Our favorite browser language allows us to provide all of the dynamic and interactive bits that we might not have if we're serving plain HTML without it.
The best example and implementation: **React, Vue.js**
They ultimately render to HTML with JavaScript right in the browser (or server) making the process of building interactions or serving dynamic content more natural and declarative.



```
class HelloMessage extends React.Component {
  render() {
    return (
      <div>
        Hello {this.props.name}
      </div>
    );
  }
}

ReactDOM.render(
  <HelloMessage name="Taylor" />,
  document.getElementById('hello-example')
);
```

```
Hello Taylor
```

Figure 16: JavaScript (React) [9]

The code which is above is called JSX. In other words, it is a connection between JavaScript and HTML. Those HTML files include a group of assets like images, and CSS.

API

This is why all applications, which are using JAMStack idealogy, are dynamic. Your application will use Javascript to make an HTTP request to another provider which will ultimately enhance the experience in one form or another.
The best example and implementations: **StoryBlok, GraphCMS, and Strapi**

Markup

For markup usually is taking one framework which will render your final HTML and connect it with API and JavaScript. Static site generators and web frameworks such as Gatsby and Next.js extend those frameworks providing the ability to more easily manage the content that gets added to the projects and render pages out to static files.

5.1. What's the difference compared to traditional architecture?

If you look at the beginning of the web, we can see Jamstack concepts in practice. Before server-side solutions were in place, developers wrote HTML by hand and served static sites to their visitors.

But as the web grew, the technology matured, seeing complex and powerful server-side solutions come into the picture. That led to projects like WordPress, where with a small amount of work to install it on a server, you can be up and running with an entire website and a content management system (CMS).

While WordPress is in fact still pretty much dominant, developers wanted to deliver static content in a way that would be more performant. They were also looking for fewer moving pieces and less maintenance than typical server solutions that run on the client-side. [8]

So, inspiration for JAMStack architecture was taken from WordPress. And only to replace this technology "dynamic" architecture was created. Below you can see how architecture are working in traditional CMS and in JAMStack.

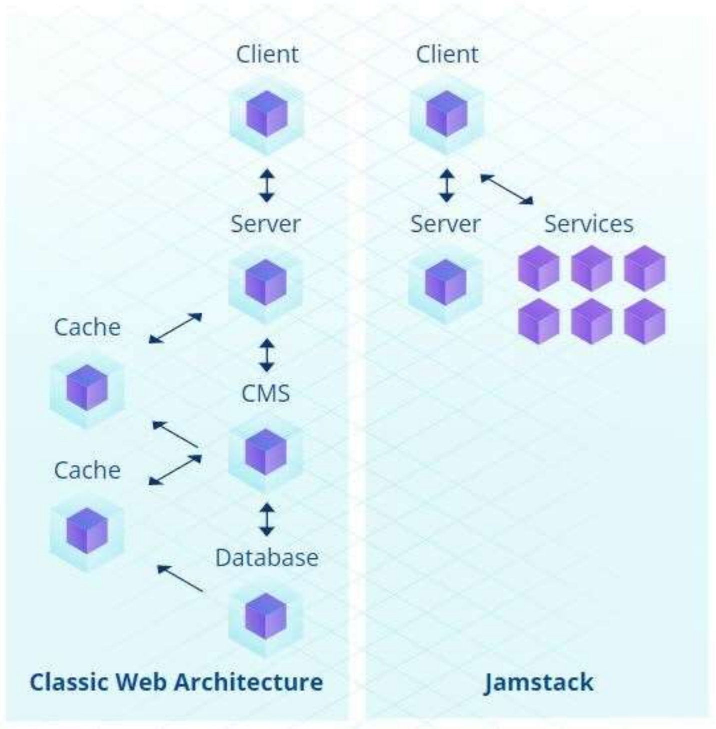


Figure 3: Implementation of solutions traditional
CMS and JAMStack architecture [10]

On the picture above, there are 2 architectures: Classic Web and JAMStack. In Classic Web we are transferring data from top of hierarchy - Database, to the lowest level - Client. There are a lot of levels between them and that is why sometimes Classic Web sites are loading really slow, because of waiting for transferring data from top level to the lowest. At the same time, JAMStack is about connecting Server to client directly and we do not need to connect server with CMS, Cache, Database. Other data is staring in “Services”, which make web applications so fast.

5.2.What are the benefits of the Jamstack?

The Jamstack comes with a lot of benefits compared to the more traditional web development approach. For instance, JAMstack web apps are fast, reliable, scalable, and usually pretty cheap.

For example, there are a few benefits, which are much better in comparison with CMS WordPress [8]:

- Limiting the number of moving pieces required to deliver that app
- Exposing fewer attack surfaces for bad actors

- Providing an app that will infinitely scale, suffering from less downtime
- Delivering the website fast as its mostly static files
- Spending less money on hosting

Moreover, there are much more possibilities to extend your benefits. For example, you can create a web application using the framework Next.js or Gatsby within 1 minute. The same time will take for deploying the web application on test server Vercel, for example. It is convenient for developers from the side of time reduction of work time and showing results for clients. Web services are also being driven by the Jamstack growth to provide a better developer experience.

While it’s already fun and easy to build web apps in the Jamstack world, it’s still relatively young. There’s a ton of growth potential and room for improvement that will continue to change how we develop web applications. [8]

5.3. **How has the Jamstack evolved over the years?**

The rapid increase of popularity JAMStack gained in 2020 when a lot of “players” of WordPress tried a new approach for creating dynamic web-application instead of static websites. In 2021, for most prospective developers it seemed that WordPress fades into the background. Only because WordPress has been on the website market for a very long time, it "remains afloat". Over time, JAMstack will completely displace the competitor.

6. Project on JAMStack architecture

My project is based on real customers' needs. A young Italian woman asked our digital agency to create a web application for her. As we are progressive developers, we decided to do it with a help of JAMStack architecture.

So, my stack is **JavaScript** - React.js; **API** - StoryBlok; **Markup** - Next.js. For deployment I used Vercel. In further chapters, I will proceed with this technology stack. Let's go to the details and describe why I have chosen exactly these technologies.

Links to client's web-application can be found in Appendix

For the basement of the project I was using "starter", where is included all necessary functions for starting the developing process.
Link to starter - https://github.com/sakyra-yp/dola-next-app-stack

"Dola next app" was developed for helping developers to create the initial structure for the projects, which will be using Next.js and others related to this framework technologies. Thanks to Dola Service - Digital Agency, we can use useful software for developing web applications.

There are included such extensions:

- initial folder structures (components - */ui, */common, styles)
- webpack (module bundler, its main purpose is to bundle JavaScript files for usage in a browser, yet it is also capable of transforming, bundling your project)
- installed Next.js and all functions as main Markup (JAM architecture)
- next-SEO - is using for promotion web-application in google searches
- react/react-dom - initial dependencies as main JavaScript (JAM architecture)

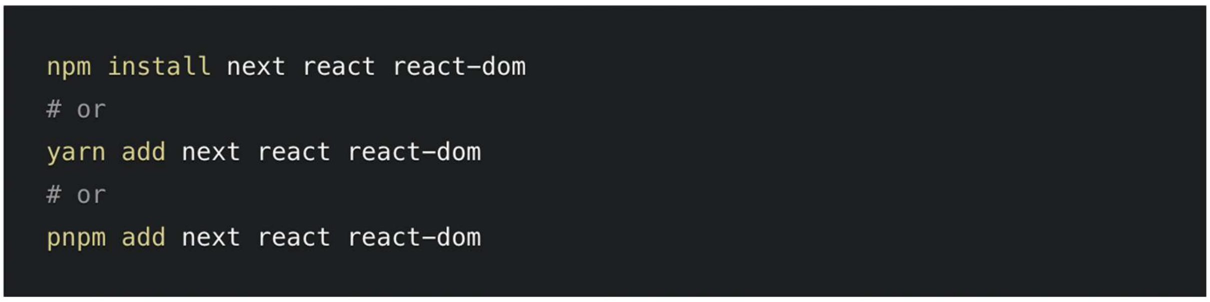
Advanced extensions:

- TailwindCSS - complicated CSS library, helps in styling
- Prettier - for formatting code
- PostCSS - is a tool for transforming styles with JS plugins.
- manual written function "cmp" for creating structures for new components with special importing routes

For storing all data and the current version I was using GitLab. The best solution for dual programming and hosting web applications. After the development process, you can join from git technologies to Vercel to make it available on the internet.

Preparation is done. Now, we are moving to installation.

As it is written in Next.js Documentation, we should install all dependencies, which were mentioned above.



```
npm install next react react-dom
# or
yarn add next react react-dom
# or
pnpm add next react react-dom
```

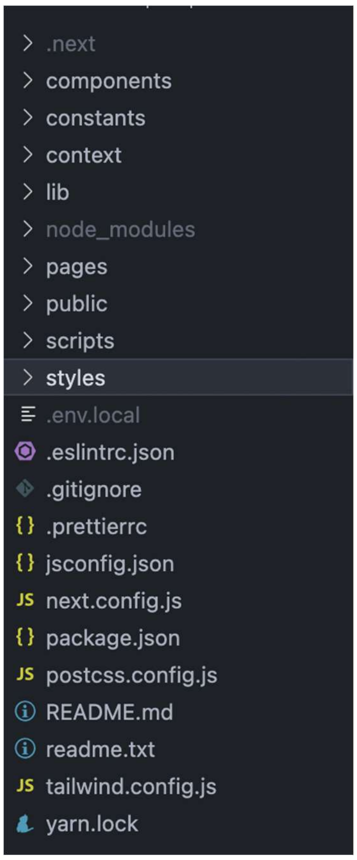
Figure 4: Next.js installation dependencies [11]

After that, start the development process. It divides into several steps:

- Developing components using *.jsx and *.module.css
- Gathering pages and creating their files in a folder
- Connecting all pages with API, in that case - Storyblok
- Pushing all files to Git technology - GitLab
- Automatically it will be pished to Vercel domain and you will get a link with your code

6.1. Project Structure

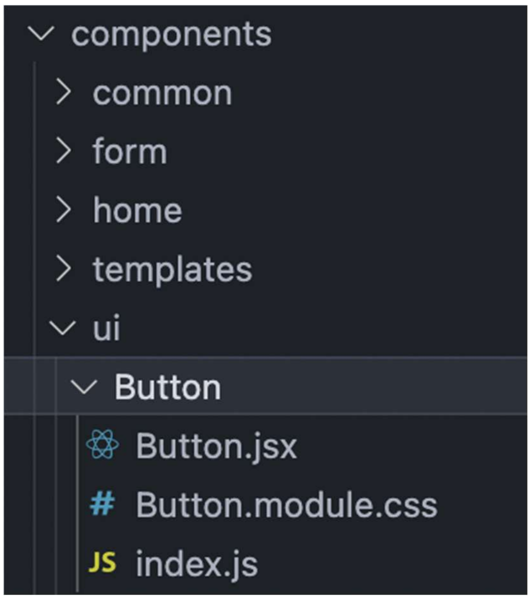
Now, I would like to show how my practical project is builded. It is a big benefit to have a look on that, how everything is developed. It is important to remain clear structure to provide perfomance and great orientation in project structure. From the point of view of developer, it is nessecary approach to remain structure.
This is how the structure of the promo code web application is looking like:



```
> .next
> components
> constants
> context
> lib
> node_modules
> pages
> public
> scripts
> styles
  .env.local
  .eslintrc.json
  .gitignore
  .prettierrc
  jsconfig.json
  next.config.js
  package.json
  postcss.config.js
  README.md
  readme.txt
  tailwind.config.js
  yarn.lock
```

Figure 5: Basic structure development

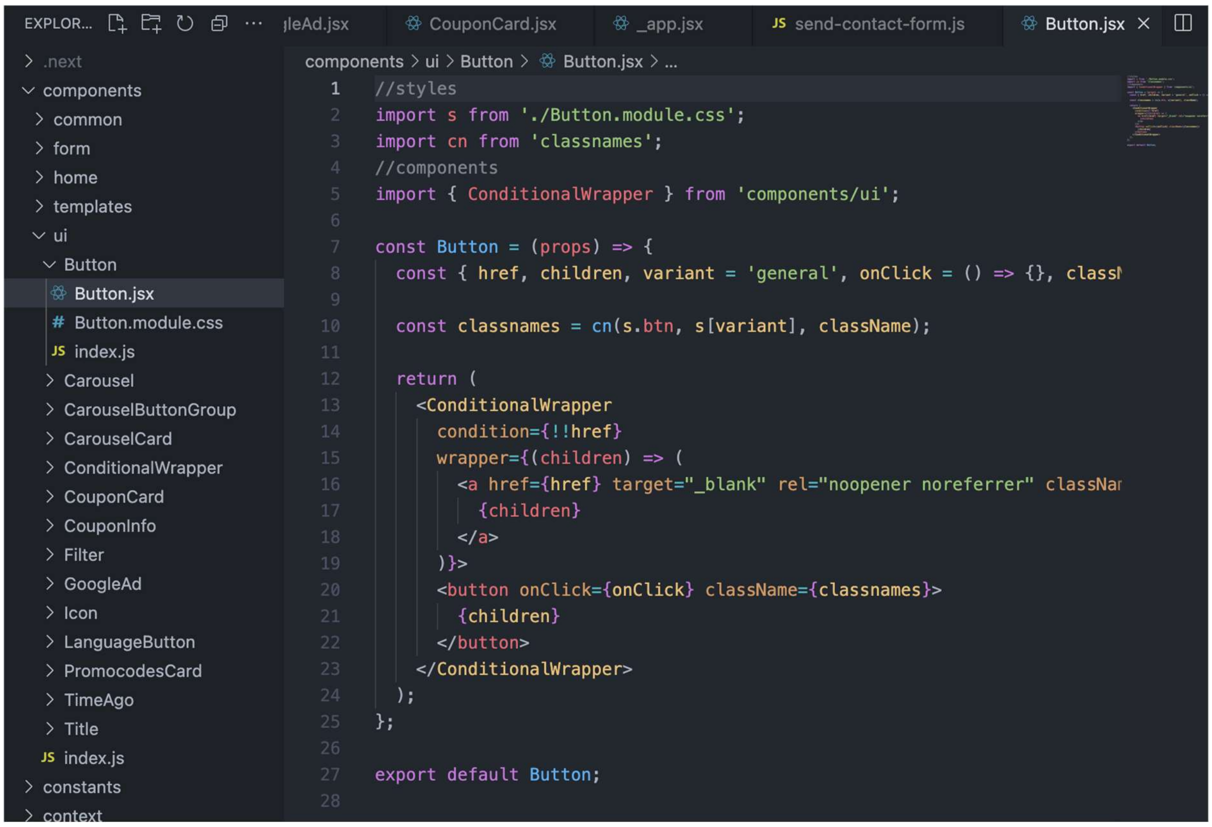
- The first folder is .next, where you can find not-important files for starting the project.
- The next folder is components, which stores all custom re-usable components.



```
components
  common
  form
  home
  templates
  ui
    Button
      Button.jsx
      Button.module.css
      index.js
```

Figure 6: File structure of component "Button"

For example, the Button component was developed for custom buttons in a web application. Button.jsx - where HTML code is stored, Button.module.css - styles applied, and index.js for re-export of component (better implementation of import in future).

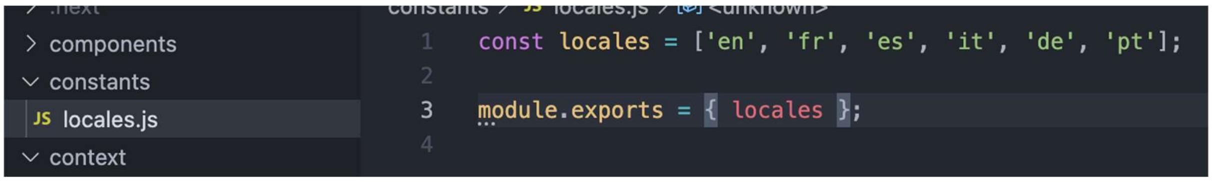


```
.next
components
  common
  form
  home
  templates
  ui
    Button
      Button.jsx
      Button.module.css
      index.js
    Carousel
    CarouselButtonGroup
    CarouselCard
    ConditionalWrapper
    CouponCard
    CouponInfo
    Filter
    GoogleAd
    Icon
    LanguageButton
    PromocodesCard
    TimeAgo
    Title
    index.js
constants
context
```

```
//styles
import s from './Button.module.css';
import cn from 'classnames';
//components
import { ConditionalWrapper } from 'components/ui';

const Button = (props) => {
  const { href, children, variant = 'general', onClick = () => {}, classN

  const classnames = cn(s.btn, s[variant], className);

  return (
    <ConditionalWrapper
      condition={!!href}
      wrapper={(children) => (
        <a href={href} target="_blank" rel="noopener noreferrer" classNa
          {children}
        </a>
      )}>
      <button onClick={onClick} className={classnames}>
        {children}
      </button>
    </ConditionalWrapper>
  );
};

export default Button;
```

Figure 7: Coding Button component

- constants folder, where we are storing all locales (languages), which should be applied and used in web applications.



```
const locales = ['en', 'fr', 'es', 'it', 'de', 'pt'];

module.exports = { locales };
```

Figure 8: Array, where all locales are stored

Unfortunately, it is not dynamic. In another word, the customer (end-user) can not add more languages by himself. This problem is due to special export which should be imported to next.config.js, where settings for the project are applied. Will be mentioned in the Recommendations part.

- Farther folder is context, where we store some data for Modals (Pop-up window). Functionality is reminiscent of Redux technology. It was created with the same structure as the mentioned example.
- lib folder is responsible for manual internal functions. Mostly, they are linked with API, in other words, it works with the "back-end" of our web application. For example, "filter-by-name.js" for correct filtering or "api/config.js" for the correct combination of frontend and backend.
- node_modules automatically generates at the moment of installing all dependencies of the project. Especially, there stored all functions of those dependencies.
- Most important folder - pages.

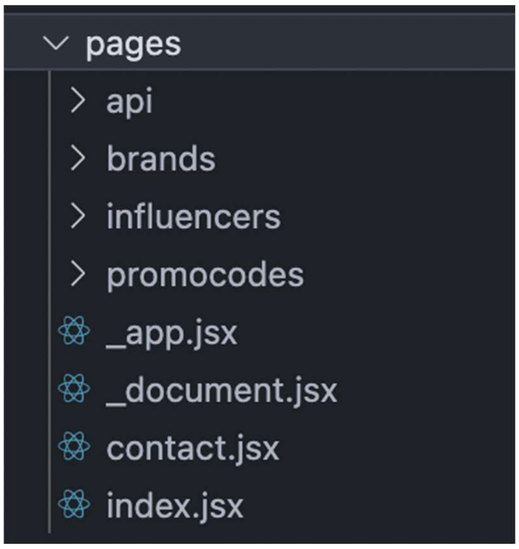
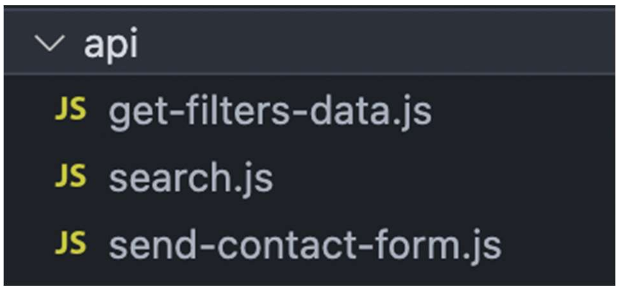


Figure 9: pages folder

There are storing all pages, which was developed, except "_app.jsx" and "_document.jsx". Those pages are automatically generated while creating the project. For example, if we enter after https://www.thedealbooster.com/ "contact" URL, we will redirect to the contact page, the code for which developed in the "contact.jsx" file. An exception to the rules is "index.jsx" - this is a reserved name for React. If we open the home page of the web application - https://www.thedealbooster.com/, Next.js will use the "index.js" file.
Also, inside of pages folder is located the internal folder api.

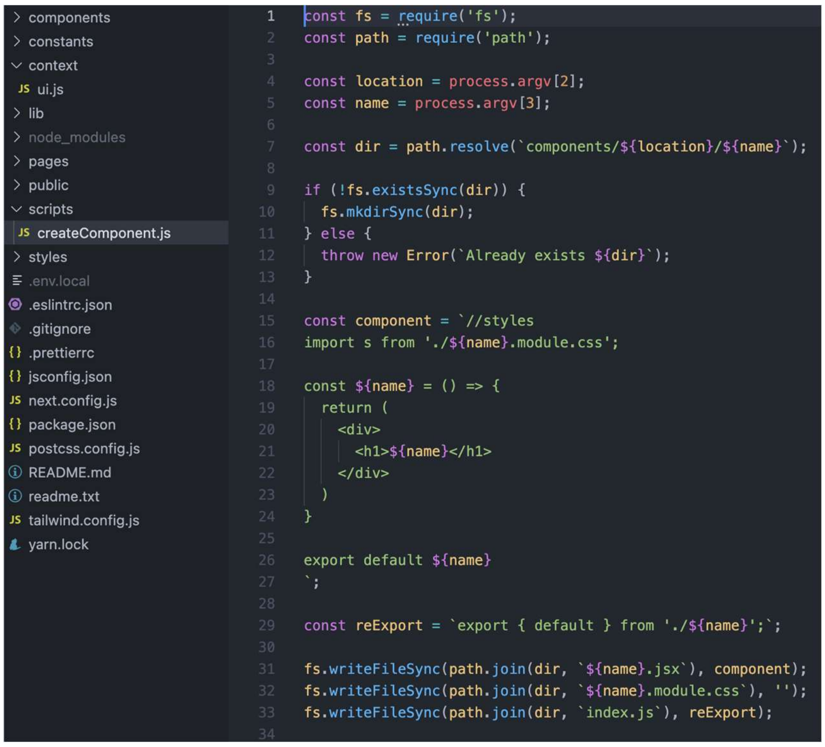


```
api
  get-filters-data.js
  search.js
  send-contact-form.js
```

Figure 10: API requests to the backend

There are located files, which send requests using the method POST to provide information to the end-user. All names speak for themselves.

- The next folder is public. There are storing static files, for example, fonts, images, favicons, and others.
- scripts custom folder, where we store script. The function is to create the structure for components automatically. For that, you need to enter in console "yarn cmp ui Button".

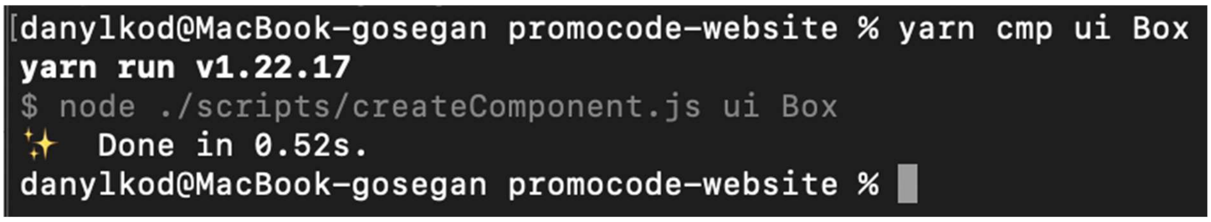


```
> components
> constants
v context
    ui.js
> lib
> node_modules
> pages
> public
v scripts
    createComponent.js
> styles
  .env.local
  .eslintrc.json
  .gitignore
  .prettierrc
  jsconfig.json
  next.config.js
  package.json
  postcss.config.js
  README.md
  readme.txt
  tailwind.config.js
  yarn.lock
```

```javascript
const fs = require('fs');
const path = require('path');

const location = process.argv[2];
const name = process.argv[3];

const dir = path.resolve(`components/${location}/${name}`);

if (!fs.existsSync(dir)) {
  fs.mkdirSync(dir);
} else {
  throw new Error(`Already exists ${dir}`);
}

const component = `//styles
import s from './${name}.module.css';

const ${name} = () => {
  return (
    <div>
      <h1>${name}</h1>
    </div>
  )
}

export default ${name}
`;

const reExport = `export { default } from './${name}';`;

fs.writeFileSync(path.join(dir, `${name}.jsx`), component);
fs.writeFileSync(path.join(dir, `${name}.module.css`), '');
fs.writeFileSync(path.join(dir, `index.js`), reExport);
```

Figure 11: createComponent.js - script for creating new comp

First position - “yarn”. It is something like “npm”, in other words, it is the standard package manager.
Secondly - “cmp”. Name of the script, nothing else.
Thirdly - “ui”, is the name of the folder where we should create the component
Finally - “Button” name of the component.



```
[danylkod@MacBook-gosegan promocode-website % yarn cmp ui Box
yarn run v1.22.17
$ node ./scripts/createComponent.js ui Box
✨  Done in 0.52s.
danylkod@MacBook-gosegan promocode-website %
```

Figure 12: command line for
creating new component

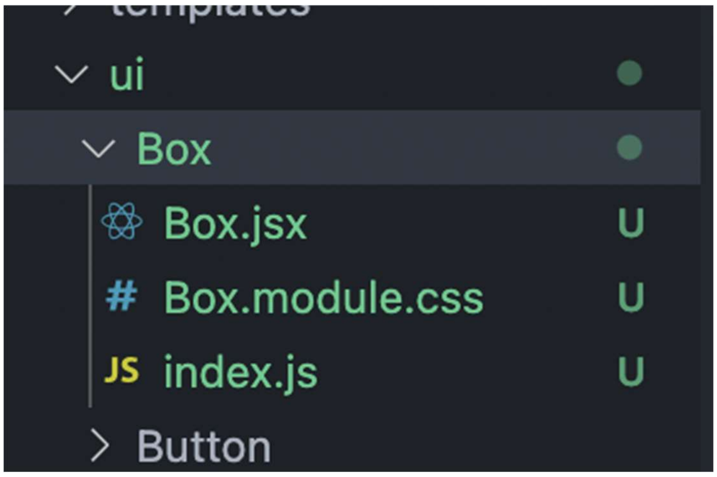
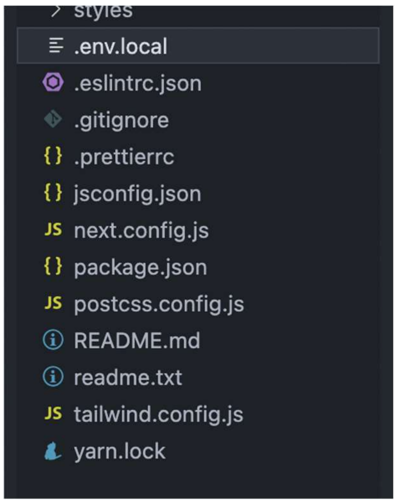


Figure 13: new component in
folder structure

- Next folder is styles. This a storage for global styles, which are applying to all pages.

- Also, there are a lot of additional files, which help us in setting up the project. Picture below will show those files.



```
> styles
.env.local
.eslintrc.json
.gitignore
.prettierrc
jsconfig.json
next.config.js
package.json
postcss.config.js
README.md
readme.txt
tailwind.config.js
yarn.lock
```

Figure 14: other files in project structure

The most important files are “next.config.js”, “tailwind.config.js” and “package.json”. In next.config.js we are setting up locales and creating paths to images and other settings. In tailwind.config.js we congifure basic styles and create custom classes. And in package.json

7. Evaluation and comparison JAMStack architecture and architecture solutions CMS

This chapter will be about comparison and evaluation of two different architecture solutions. JAMStack and CMS were created in different times. To investigate which tool is better to use nowadays, we need to compare both due to their possibilities.

To begin with, both of them are modern architectural solutions. But only one of them, JAMStack, is new, so people don't understand for now whether it is worth it for them to change architecture. Will it be profitable for them? Or maybe it will make the situation worse? Let's dig into the details.

To be more precise, you can have a look at the table below, which shows us all the advantages and disadvantages of technologies. This table is prepared based on information which is in open source. It is shown which way of thinking is better in different evaluation criterias.

There are some criterias defined by developers, that they use to give evalution of web-site at all. Many of them are based on technical points of view.

-	JAMStack	Traditional CMS
Performance	Superior	Better than JAMStack
Code reusability	High chance	No chance
SEO Benefits	Yes	Need additional efforts
Page Load Speed	Fast	Slow
Security	High	Low
Time for developing	Little time	A lot of time
Cost	Low	High
One environment	No	Yes
Dynamic and flexible	Yes	No
Pain points	No	Yes

From the side of perfomance Traditional CMS is better than JAMStack, because CMS has easier approach for editing data on such web-sites. At the same time, reusability much better in JAMStack, because of components, which can be used in different places. And for each such place is pulling up same piece of code. SEO optimization in JAMStack is great way to promote web sites. It is additional service, information about was above in comparison table. But in CMS we need to connect it with plugin, which create bigger amount of data in databases. Due to reasons above, from point pf page loading, faster is JAMStack architecture. Respectively, time for developing costs will be higher in CMS than in JAMStack. But from the perspective of environment it is clear, that for JAMStack it will not work. Plugins for example, in CMS are "variables" from the same environment, but in JAMStack you are connecting such extensions from other sources.

Most of criterias have been taken from my own experience, which is proved in my practical part. Also, as a confirmation of my words, you can have a look on two web-sites which were created with 2 different approaches:

https://www.whitehouse.gov/administration/ - web-site based on WordPress
https://www.thedealbooster.com/ - web-application based on JAMStack approach

Also, there is a different approach in rendering pages for both of them. For JAMStack the main feature is Git technologies from where everything is deployed. While for traditional CMS, you need to buy a separate domain, where you need to store files. Sometimes, it is time-consuming to get in touch with other platforms, often even with international domains with foreigh language.

Another case is traditional CMS. For each click in the navigation menu, web-site is rendering all web-site, while JAMStack is rendering only exact parts for users, which need to be shown.

Below you can find a scheme, how workflow is working for both of the architectures.

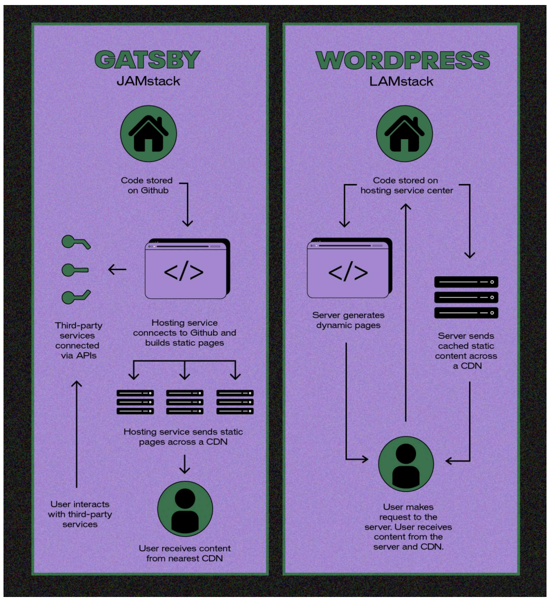

GATSBY
JAMstack
Code stored on Github
Hosting service conncects to Github and builds static pages
Third-party services connected via APIs
Hosting service sends static pages across a CDN
User interacts with third-party services
User receives content from nearest CDN
WORDPRESS
LAMstack
Code stored on hosting service center
Server generates dynamic pages
Server sends cached static content across a CDN
User makes request to the server. User receives content from the server and CDN.

Figure 15: Comparison workflows of architectures

To sum up, all information about JAMStack architecture and example of traditional CMS - WordPress, I prepared small list of cons and pros:

JAMStack:

- JAMstack web-applications are faster than dynamic pages (pros)
- The site is safer and more resistant to hacker attacks (pros)
- High scalability thanks to the use of the CDN network (pros)
- Ease of maintenance and service for programmers (pros)
- Super SEO support, moreover, it prioritized by Google (pros)
- You need to know JavaScript, API, Markup to build and maintain a website (cons)
- You will need to rebuild the page even with a slight making change (cons)

WordPress:

- Intuitiveness and ease of use (pros)
- Simple installation on the server (pros)
- If your web-site well optimized, there are search engine friendly (pros)
- The necessity to take care of safety (cons)
- Possible performance issues (cons)
- Low page speed may cause SEO problems (cons)

Lastly, there are 3 situations:

- a small company that is just getting started
- fast growing startup or a B2B business
- newsletter company (web-site with a lot of number of interactions and requests for backend)

For the first two cases, it is clear that we can use JAMStack architecture. Because we can see that there are small companies and requirements for web-site will be the same. Small, fast with safety web-application. Some eCommerce in the 2nd case is also possible.

But for the third case, we will need to use a huge backend, that is why it will be better to create this web-site on a traditional CMS - WordPress. Or there is another possibility - to hire a backend developer, which will code the “back” part for your web-site. But it will be more money consuming in that case

8. Summary of the bachelor's work

Summing up, this paper has shown about two modern architectures on the web. Theory for both of them and information in implementation by people. Difference between them and different approaches in page rendering.

Overall, traditional CMS, for example, WordPress, has been around for a long time, and it's constantly getting better and better. What used to be a humble blogging platform has expanded into a powerful CMS that's ready to take on any type of website. But, once you learn a little about JAMStack and what it can do, you'll likely find it's a perfect fit for your needs.

This bachelor work was aimed for researching which architecture or approach is better to use. JAMstack sites are a fun part of the Javascript ecosystem. If your teammates are handy with JS, or the budget for developers, you can have a blazing fast site.

If you want to build any type of website, from a blog to an eCommerce store, WordPress is a great option. However, JAMStack showed us in a comparison table that approximately 80% of evaluation criterias is better than traditional CMS.

Finally, it seems to be that JAMStack architecture is the best solution for web sites which should be: with great performance, low cost, safety, faster, easier to scale and it solves many pain points both for website owners and developers.

9. Conclusions and recommendations

For sure, there are a lot of topics which weren't covered in this bachelor's work. For this reason, I would like to give a few recommendations for readers of these paperwork.
So, there a few conclusions and recommendations:

- Dynamic locales
  The thing is there is no possibility in practical project to enter new languages dynamically. In other words, from the Storyblok Admin Panel you can not add new language. In fact, you can, but it will not be working for you properly. The problem is the file next.config.js. There is using special imports, which should be created by module.exports.

- Creating "your account profile"
  It would be a great solution, to have in future your own profile, where can be displayed, for example, some statistics of the end-user. As it is a promo-code web-application, there can be written how many times the code was copied and etc. Also, from profile end-users can create their own promo-codes to make work easier for the main admin. In that case, we can use the BEAR token to make automatic log out after 30 days of inactivity.

- Create paid subscription
  Interesting idea, which is getting popular nowadays - VIP status. So, the thing is that you, as a user, will receive VIP status after paying some money. After, you can receive VIP coupons and promo-codes with huge discounts. Exciting marketing ploy for the owner of the web-site.

- Try to use another Stack
  Great opportunity will be to create the same web-site using another Stack. For example, JavaScript - Vue.js, API - Graph CMS, Markup - Gatsby. After that, compare both stacks of JAMStack architecture and find a better option for end user needs. This way, developer and owners can find the best stack for them.

Also, the best option is to try to use JAMStack architecture as a basis for your web application by yourself. The main idea is developers receive irreplaceable experience and pleasure to work with such modern solutions.

However, **Jamstack will not be the best solution** for you if you do not know JavaScript. Also, Jamstack **is not the best option for very dynamic sites**. Including ones that need constant updating, such as news sites or platforms with many user interactions.

In any other case, **creating a website based on Jamstack can be a great solution**. Especially in the case of industry blogs, e-commerce stores, business cards, landing pages, and sites for software companies.

10. References

1. Wright, K. (2020, November 5). *What is WordPress? | WordPress 101 Tutorials*. IThemes. https://ithemes.com/tutorials/what-is-wordpress/
2. Kinsta. (2021, April 27). *What Is WordPress? Explained for Beginners*. https://kinsta.com/knowledgebase/what-is-wordpress/
3. Schaffer, N. (2021, August 25). *The 17 Most Useful WordPress Tips and Tricks to Blog Like a Pro!* Social Media & Influencer Marketing Speaker, Consultant & Author. https://nealschaffer.com/wordpress-tips-and-tricks/
4. WordPress vs Other Content Management Systems - Namecheap. (2022). Namecheap. https://www.namecheap.com/wordpress/wordpress-vs-other-content-management-systems/
5. Google Afbeeldingen. (2022). Google Images. https://images.google.com/
6. S., J., T., T., J., & D. (2022, January 25). WordPress Shortcodes Plugin — Shortcodes Ultimate. WordPress.org Česko. https://cs.wordpress.org/plugins/shortcodes-ultimate/
7. Wright, K. (2020, November 5). What is WordPress? | WordPress 101 Tutorials. iThemes. https://ithemes.com/tutorials/what-is-wordpress/
8. Fayock, C. (2022, March 31). *The Jamstack in 2022: Why (and how) to get started*. Snipcart. Retrieved April 20, 2022, from https://snipcart.com/blog/jamstack
9. Fayock, C. (2020, June 8). *What is the jam stack and how do I get started?* freeCodeCamp.org. Retrieved April 20, 2022, from https://www.freecodecamp.org/news/what-is-the-jamstack-and-how-do-i-host-my-website-on-it/

10. JAMStack architecture. (2022) *For fast and secure sites: Jamstack*. Jamstack.org. (n.d.). Retrieved April 20, 2022, from https://jamstack.org/

11. *Wordpress vs JAMstack*. (2022). We Are Capicua. https://wearecapicua.com/wordpress-versus-the-jamstack/

12. JAMStack architecture. (2022) *For fast and secure sites: Jamstack*. Jamstack.org. (n.d.). Retrieved April 20, 2022, from https://jamstack.org/

Appendix

Appendix A

Analysis of experience with WordPress usage

According to statistics (chart below), WordPress powers over 60 million websites, or 36% of all websites on the internet, and over 60% of all websites whose content management systems are known. Because WordPress is free and so easy to use, it has been widely adopted as the gold standard for website builders.

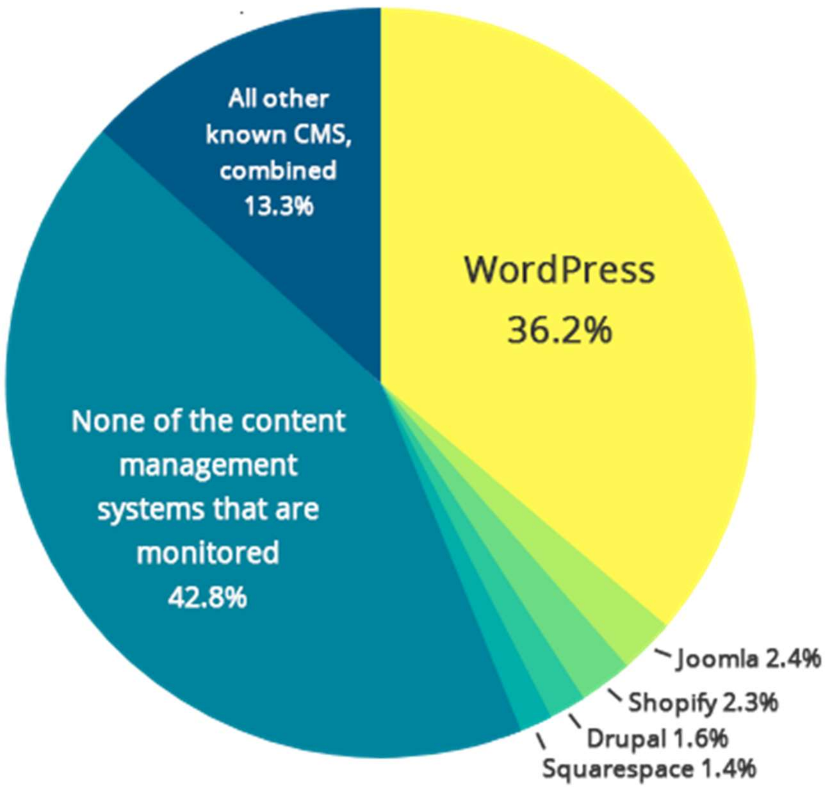


Figure 2: Comparing percentage of CMS [7]

5.1. Big companies, that are using WordPress

CMS WordPress is used by people all world. More than 150 big companies are applying this Content Manager System to their website. Mostly, are built with their own templates.

WordPress plugins

Plugin is part of code, which helps user solve different tasks. Users can use them for all their needs. It is useful when the customer is not advanced in coding in PHP. And instead of spending a lot of time finding some code, you can solve the problem with one click and basic setup.

Plugins can be added via the Plugins menu in your WordPress dashboard. Using the built-in search function, you can find a free plugin available on the WordPress.org Plugin directory. Plugins can also be packaged as zip files that you upload through the Plugins menu uploader in your WordPress dashboard.

Empiric evaluation of WordPress usage

Our research's main goal was to define evaluation criteria and evaluation methods. Also, how well educated and informed people are while using WordPress. Firstly, the analysis of WordPress forums had to lead us to come up with the following Hypotheses:

13. People are not enough educated to build or create their own website on WordPress
14. Developers on WordPress are not using all of the WordPress functions
15. Developers are not using web-sites constructors to build good websites
16. Plugins on WordPress are a very popular thing in WP and take a lot of time to get into details

To extract the most comparable data for us and make our research as realistic as possible was decided to take a deductive approach. We have chosen the form of the questionnaire because we wanted to conduct our research on students. Our sample size is 47 and the approach we took gave us the possibility to collect both Quantitative and Qualitative data.

The platform on which we conducted the survey was google/forms.com because we had to make our questionnaire as complete as possible regarding the number of questions. We decided to choose the English language for our survey to cover the common tongue spoken among our sample data. The survey was posted on social media such as Instagram and Facebook and various private university chat groups. Our survey consisted of 11 questions. The first two questions were the general data collection on the developer respondents' wellbeing. The next six following questions were to understand how deep their knowledge and skills about WordPress. The last question is aimed to get statistics about building a website with its own code or constructor.

Questionnaire Analysis

We derived from our questionnaire that about 38.3% of our respondents are aged 17 to 20, 31.9% range from 21 to 24, and the rest are older. When we look at the developers' well-being, the survey indicated that the majority of them consider themselves experienced creators of websites, as over 75% of the respondents see themselves as either 'Intermediate' or 'Advanced'. In other words, more than 75% are working with WordPress for at least one year. What was surprising for me, was that 14,9% seem to themselves as professional developers with background experience of 10 years.

If we refer to figure 9, we investigated that the most popular usage of WordPress is for Blog websites, which is 61,7% of all lists. Also, we found that 52% of developers are using WP for Landing pages and for Portfolio websites; 42,6% of the respondents had admitted that Social media and E-commerce websites are hard to create without CMS WordPress. However, for me surprisingly, that Magazine website (31,9%) turned out to be less than Business websites (38,3%).

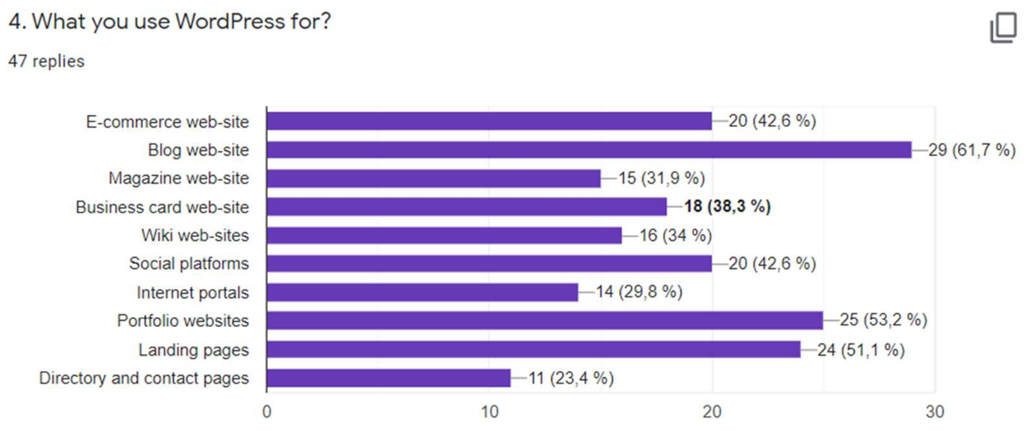
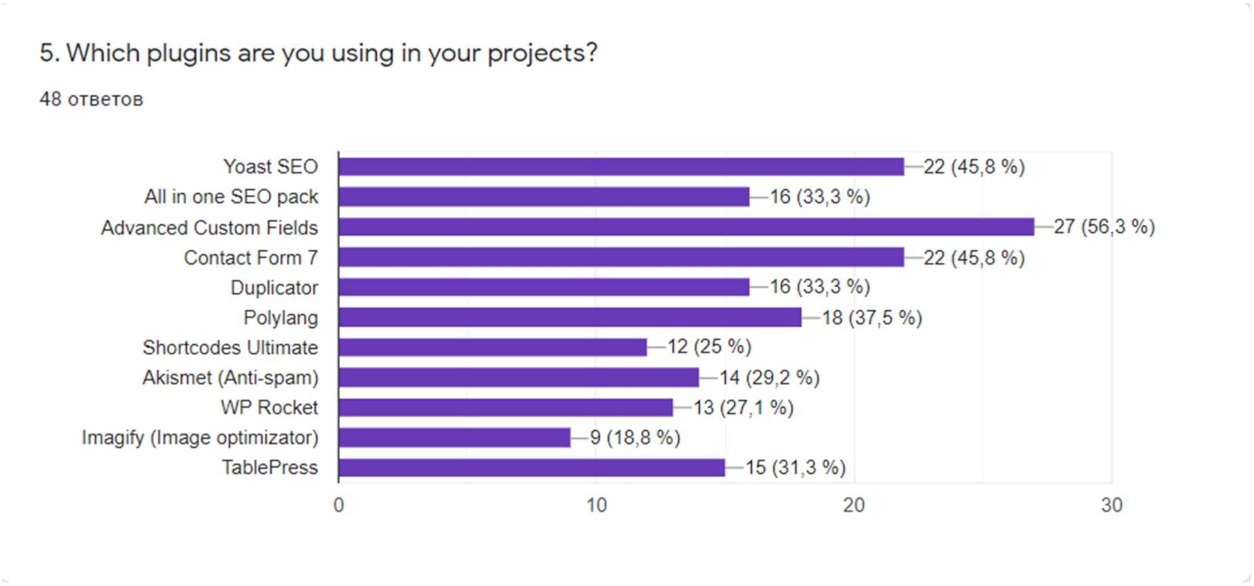


Figure 9: Main results of the questionnaire analysis. (the author‘s own work)

By analyzing the survey results, we looked at the questions that were intended to give us a better understanding of the developer’s interests. The chart below shows us the most popular plugins among WordPress developers. As you can see, the most popular is Advanced Custom Fields, a plugin that helps us to make a content dynamic (56,3%). 45,8% of respondents stated that they prefer to use plugins such as Contact Form 7, which helps us collect data from websites, and Yoast SEO which allows us to promote your website in different searches; 37,5% of the respondents had admitted that the best plugin for them is Polylang. It helps us translate your pages to several languages at the moment with help of a switch. Not a little important plugin is TablePress, Duplicator, and Shortcodes Ultimate, which are using 31.3%, 33.3%, and 25% respectively

Moreover, referring to Figure 11, we can reinforce this idea as 68.8% agree that they are taking information about WordPress features from YouTube. 64.6% of respondents take



information from different internet forums. The famous one is Stackoverflow. Or another way of using such kind of information is just googling your problem or question. More than half of respondents stated that they prefer to take information from Documentation of WordPress or PHP and from their colleagues and friends. Other ways of taking information are University or high school and Internet or offline courses. Moreover, only one person from 42 voted for the discord server, where users can ask a question to help with solving problems or questions.

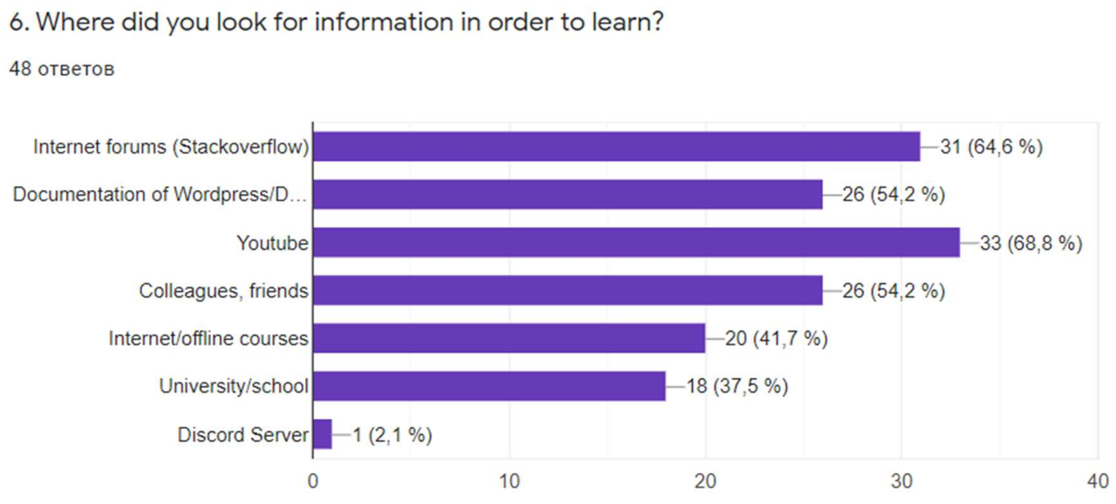


Figure 11: Main results of the questionnaire analysis. (the author‘s own work)

Referring to figure 12, there we have 2 interesting questions. The first chart shows us that majority of developers are using free plugins, which means that the functionality of free plugins is enough for them.

On the second chart, we can have a look that 75% of respondents don’t know and don’t use all of the functions in WordPress. This means that there is a lot of superfluous in WordPress.

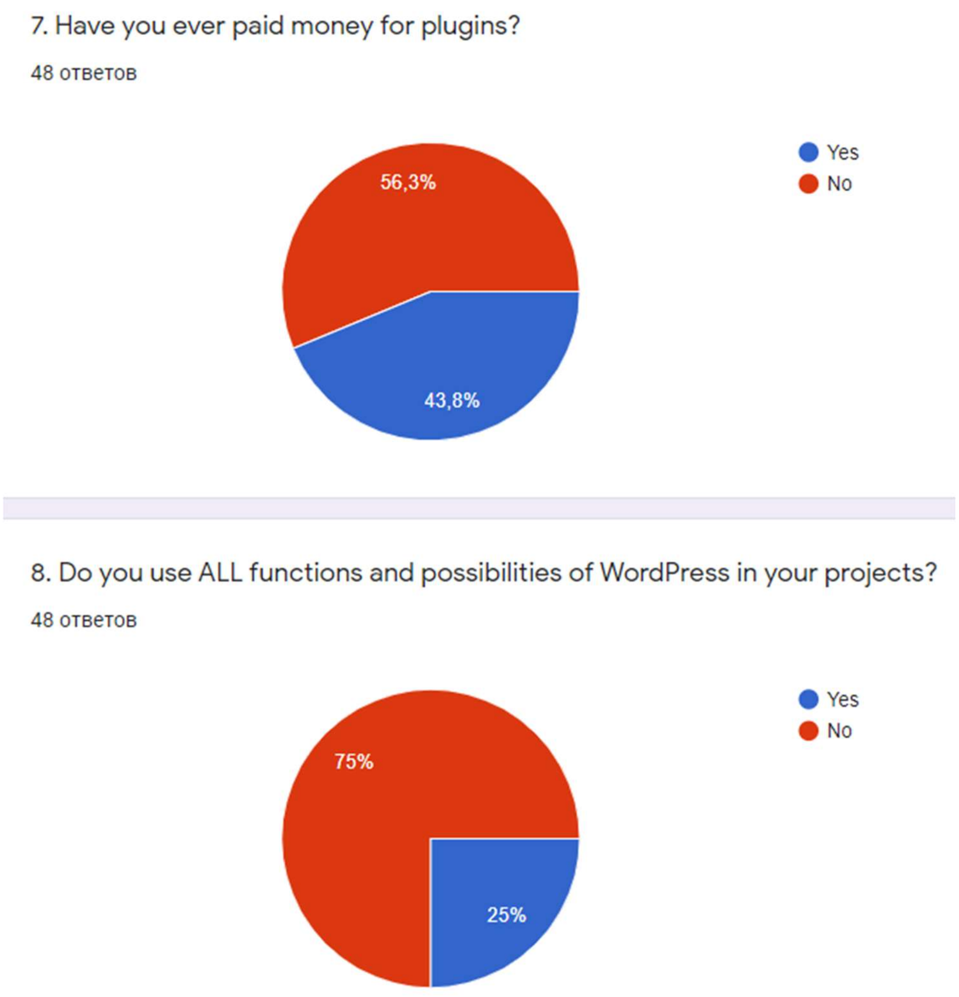


Figure 12: Main results of the questionnaire analysis. (the author‘s own work)

By analyzing survey results from the last question we can make a decision that the majority of developers are using constructors very rarely. 27.1% are never using it, but 35.4% of them use it sometimes. So, that kind of respondent prefers to create websites by own code. Although, 20.8% often use constructors as Elementor. And 16.7% of respondents are using it permanently.

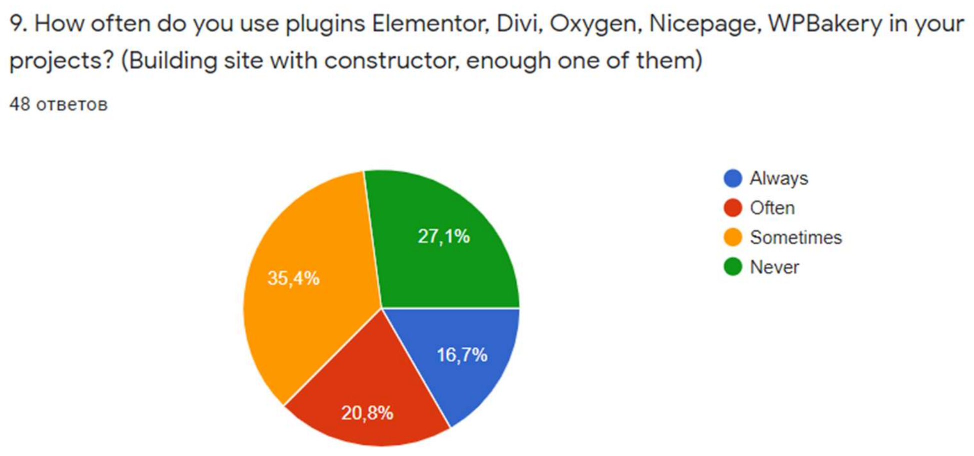


Figure 13: Main results of the questionnaire analysis. (the author‘s own work)

Conclusion

Overall, our investigations of our sample showed that most developers consider themselves professional creators. And their experience is 1-3 years mostly, which is enough to create a website. It our first Hypothesis, “People are not enough educated to build or create their own website on WordPress”.

Our questionnaire analysis also proved our second Hypothesis, “Developers on WordPress are not using all of the WordPress functions”.

However, our third Hypothesis, “Developers are not using web-sites constructors to build good websites.” Is false, as our finding indicated that not all developers are not using constructors. And it is true, that on the internet a lot of good websites, which were built on constructors such as Elementor, Divi, Oxygen and etc.

Our fourth Hypothesis, “Plugins on WordPress is a very popular thing in WP and take a lot of time to get in details” is also false. As you saw on the charts above, developers are quite interested in plugins, because they make their work easier.

Finally, based on my own experience, there are a few disadvantages to WordPress technology. As a developer, I can say that WordPress is great when you have some template from the Internet store based on WP functions. But when it comes to coding your own template, it gets harder. You waste a lot of time setting up a project, turning on different settings, and only after that coding your parts of the code (template). Moreover, as it was mentioned above, this platform is slow (compared with JAM Stack technology) and it has sometimes problems with connecting to the database MySQL

Appendix B

Links to customer's web-application made by author

Link to Vercel deployment domain - https://promocode-website.vercel.app/
Link to the real domain - https://www.thedealbooster.com/

University of Hradec Králové Faculty of Informatics and Management Academic year: 2020/2021	Study programme: Information Management Form of study: Full-time Specialization/combination: Information Systems Management (im3-p-an)

Document for registration BACHELOR THESIS

Name and surname:	**Denys Danylko**
Personal number:	**I1900795**
Address:	Palachova 1129/17, Hradec Králové, 50012 Hradec Králové 12, Česká republika
Work topic:	Alternativy k systému pro správu obsahu pro poradenskou firmu
Work topic in English:	Alternatives to the content management system for a consulting firm
Supervisor:	Mgr. Daniela Ponce, Ph.D. Department of Information Technologies

Theses guidelines:

Aim: To compare and evaluate JAMStack architecture and architecture solution of traditional Content Management System. Outline: 1. Introduction 2. The aim of bachelor's work 3. Processing methodology 4. Traditional CMS – WordPress 5. Analysis of experience with WordPress usage 6. JAM Stack 7. Evaluation and comparison JAMStack architecture and architecture solutions CMS 8. Summary of the bachelor's work 9. Conclusions and recommendations 10. References

Recommended resources:

Student's signature: Date:

Supervisor's signature: Date: