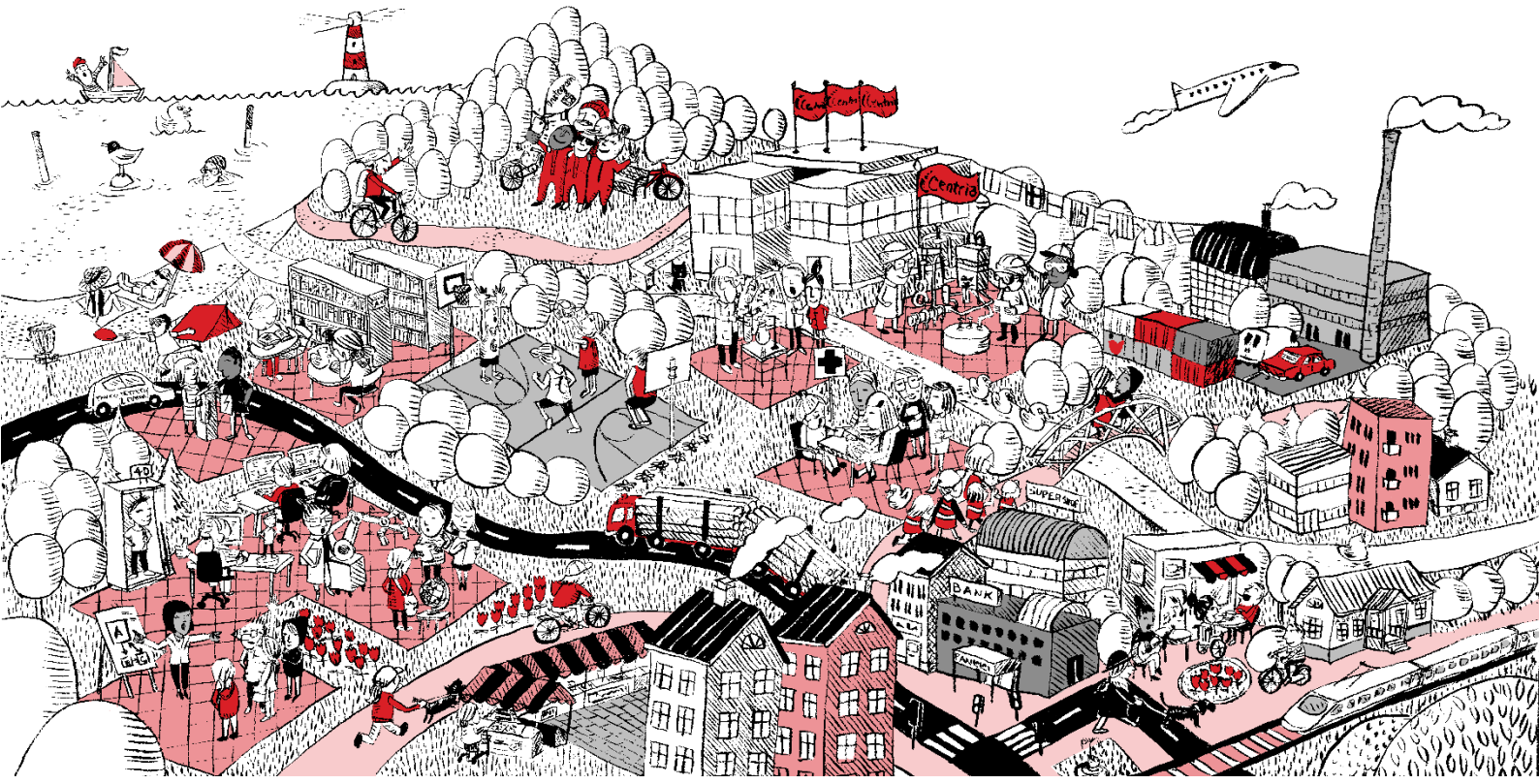
Vu Nguyen

BUILDING WEB PAGE TEMPLATES WITH GATSBY AND MARKDOWN

Thesis
CENTRIA UNIVERSITY OF APPLIED SCIENCES
Information Technology
August 2021



ABSTRACT

Centria University of Applied Sciences	**Date** August 2021	**Author** Vu Nguyen
Degree program Bachelor of Engineering, Information Technology		
Name of thesis BUILDING WEB PAGE TEMPLATES WITH GATSBY AND MARKDOWN		
Centria supervisor Kauko Kolehmainen, Heikki Ahonen		**Pages** 34
Instructor representing commissioning institution Kauko Kolehmainen, Heikki Ahonen		

The purpose of this thesis was to create a template for the course pages of Centria University of Applied Sciences. Since there is a demand of the Information Technology department for a web-based template that can help lecturers to publish their teaching documents on the internet so students and other people can access them via Github pages, this project aimed to design and build a modern template based on those requirements.

The thesis discussed two main contents. The first part takes a general view of the necessary technologies used to build the project, such as React, Gatsby, Markdown, GraphQL. The contents of this part helped to define the specifications of the project and pointed out the reasons why Gatsby was used as the main framework and its exceptional features in working with static pages, which in this case was a document template. The second part was implementing the project and necessary configurations to deploy in on Github Pages.

As a result of the thesis work, the document template was an example of using Gatsby to create and deploy a static page.

Keywords
React, Gatsby, GraphQL, Markdown, static site generator, Github pages

ACKNOWLEDGEMENTS

This thesis work was supervised by the Centria Information Technology department during the period March 2021 – August 2021.

First and foremost, I want to send my gratitude to my thesis supervisors Kauko Kolehmainen and Heikki Ahonen, also my direct manager for the project, who always supported and gave me informative pieces of advice on how to implement the project. Without their support, I could not have completed this thesis work properly. Besides, I would like to thank my family, who are always beside me whenever I need their support, and my dear friends in Finland who went through memorable years with me. And Thao Nguyen, who has always supported me and shared with me every happy as well as the difficult moment in my student life.

Turku, August 2nd, 2021

Vu Nguyen

CONCEPT DEFINITIONS

API	Application Programming Interface
CI/CD	Continuous Integration and Continuous Delivery
CSS	Cascading Style Sheets
DOM	Document Object Model
IT	Information Technology
MVC	Model-View-Controller
JS	JavaScript
HTML	Hypertext Markup Language
SSG	Static Site Generator
UI	User interface

ABSTRACT
CONCEPT DEFINITIONS
CONTENTS

FIGURE

1 INTRODUCTION

Nowadays, building a website is an essential demand for many people and organizations. Their application can be in entertainment, business, and education. The website itself can be simple as a static site where it can be used to visualize information such as blogs, portfolios, and material pages, or it can be implemented with more complex functionalities like saving and retrieving data, executing payment methods as a dynamic site. There is a tremendous number of websites on the internet. Therefore, the technologies to support and create those websites are increasing as well. This thesis gives a general view of creating a static site with the support of Gatsby and Markdown.

The purpose of the project is to help the Information Technology department of Centria University to re-build a web page template that can help lecturers generate course pages with the contents written in Markdown format then publish them on Github pages as a hosting place to help other students and users to access them easily. The template must satisfy the requirements about visualization, accessibility, and functionality, such as it should have a modern design to help users read the contents comfortably and be simple to use so lecturers can create the site's contents effortlessly.

In this project, Gatsby is the main framework because of its outstanding features in working with static sites. As the static side generator, Gatsby helps to configure the codebase with the necessary plugins and dependencies. The site's content is written by Markdown or MarkdownX. Since MarkdownX supports more features and is more compatible with component-based applications such as React, and Gatsby is also a framework of this technology. Therefore, MarkdownX is recommended for this project over the regular Markdown files. The contents of the template can be accessed by Gatsby with GraphQL as a query language. Therefore, the application does not need to care about how data is saved or managed. The combination between Gatsby and MarkdownX helps to achieve the requirements about accessibility, functionality as well as visualization of the application.

2 LITERATURE REVIEW

The first part of the thesis gives an overall view of the technologies used to build the project. Since the purpose of the project was to build a template that helps to generate static sites for course material websites, thus knowledge about technologies related to web development and content generator are necessary. The following sections give the audiences glimpses of how to develop a website with React framework and Gatsby. Besides, it also explains how Markdown helps to convert the content of a page from data files to display them on the website. Finally, the thesis gives the audience a closer look at deploying a website from its source code to host them somewhere on the internet. In particular, the thesis work shows the audience how to host a website on Github Pages, a free service offered by Github.

2.1 REACTJS

The project is based on GatsbyJS that is a framework built on top of ReactJS; therefore it is necessary to understand how ReactJS work and why the project should use it to generate a website. Nowadays, React is one of the most popular frameworks used to develop the front-end part. React is a component-based library that is designed to help to develop interactive user interface components. By incorporating the view layer (V) in MVC (Model View Controller) pattern, React enables the development of complex web-based applications which can change their properties without refreshing the whole page (Aggarwal, 2018). This leads to a better user experience with faster and more maintainable web apps development.

2.1.1 ReactJS introduction

With the rapid development of technology, web development is also constantly changing and having more requirements to meet the demand of the users. Developers need more modern and elegant solutions to implement their job more efficiently. To improve and support the developers with UI features, more frameworks and libraries have appeared and developed. Besides jQuery, Angular, and VueJS, React is one of the most popular frameworks for web development. Although still standing at the second position, after jQuery, React has been gaining popularity among developers steadily over the years (Stack Overflow, 2020). The rapid growth of React is shown in FIGURE 1.

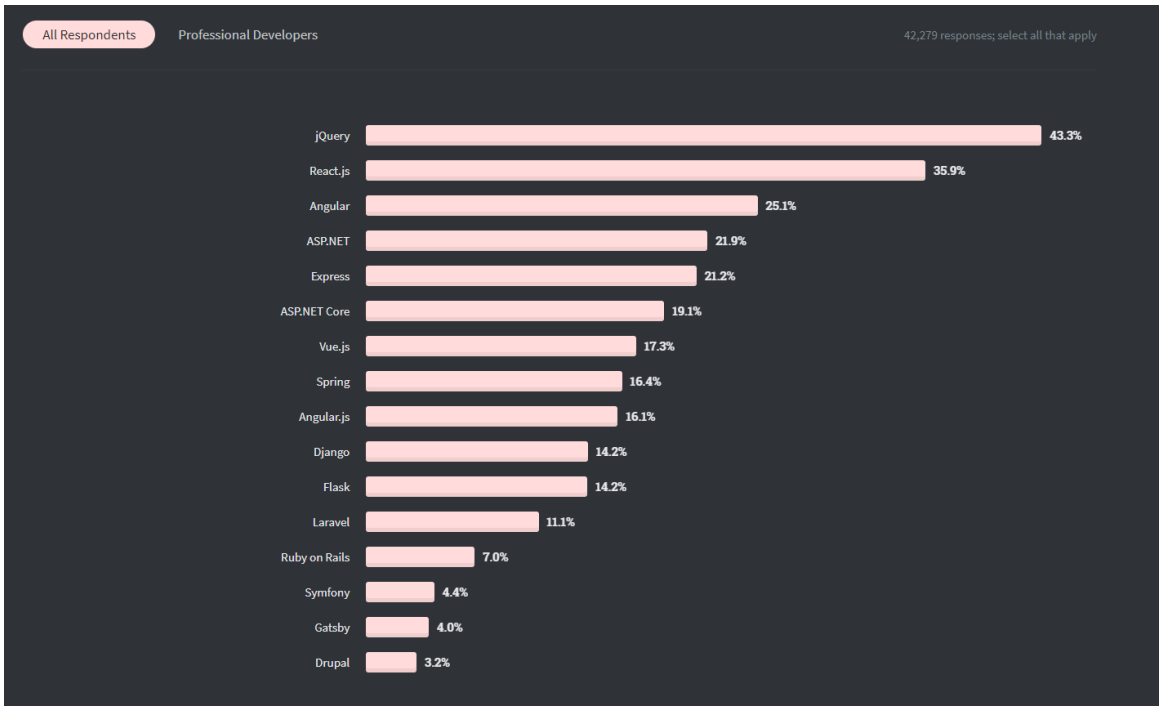


FIGURE 1. Most popular web frameworks (Stack Overflow, 2020)

React is a JavaScript framework built by Facebook to facilitate the creation of interactive and reusable user interface (UI) components. React helps to build encapsulated components that manage their state, then compose them to make more complex UI components (React Official Site, 2021). React manipulates the Document Object Model (DOM) by creating a virtual DOM (shown in FIGURE 2) and only updates the UI components, when necessary, thus offering a simple, performing, and robust application development experience (React Official Site, 2021).

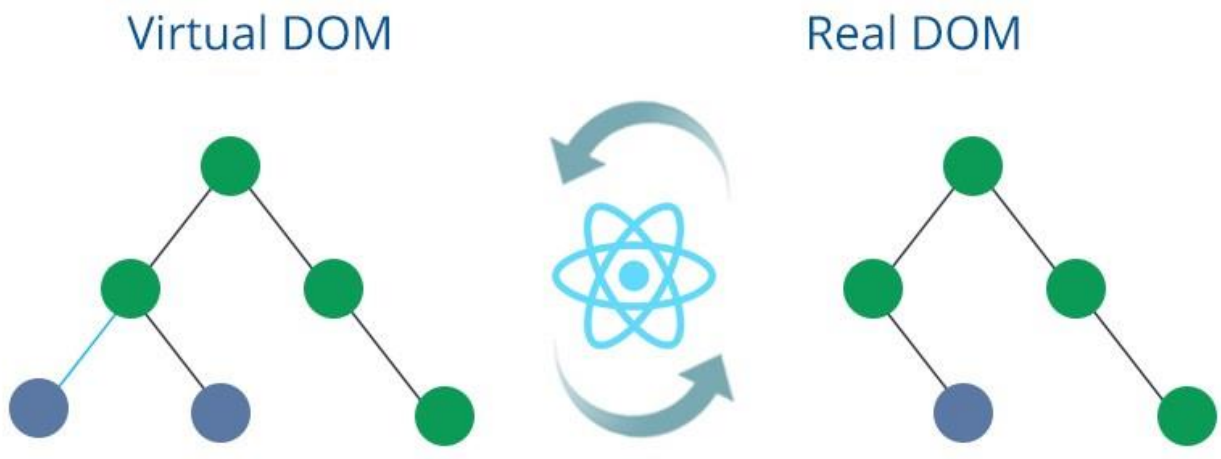


FIGURE 2. React manipulates real DOM by virtual DOM (Nguyen, 2021)

2.1.2 ReactJS features

To understand why React has become one of the most popular used by web developers, this section looks at its core features and the mechanism behind it. React supports a much efficient and lightweight Document Object Model (DOM). As mentioned before, React does not interact directly with the DOM generated by the web browsers but the virtual DOM stored in the memory. This results in the faster and smoother performance of the application. In most other frameworks and libraries, they interact with the browser DOM directly and refresh the whole page every time an event is triggered. As a result, when a significant amount of data needs to be modified, the application's performance is affected heavily. On the other hand, with virtual DOM, React only needs to compare it with the actual DOM and modify the necessary parts on the page. That makes its job more straightforward and more efficient (Aggarwal, 2018). The mechanism behind the virtual DOM is relatively simple. Whenever the application requests for changing its content, these changes are copied to the memory residing in virtual DOM first. After that, a functionality will compare the virtual DOM to the browser DOM, and then the required changes only are reflected in the browser DOM, instead of reloading the entire DOM. By doing this, ReactJS boosts the application's performance with faster speed, especially in the significant projects where thousands of changes are to be made (Aggarwal, 2018). FIGURE 3 depicts the process of comparing and updating the DOM with the help of Diff functionality.

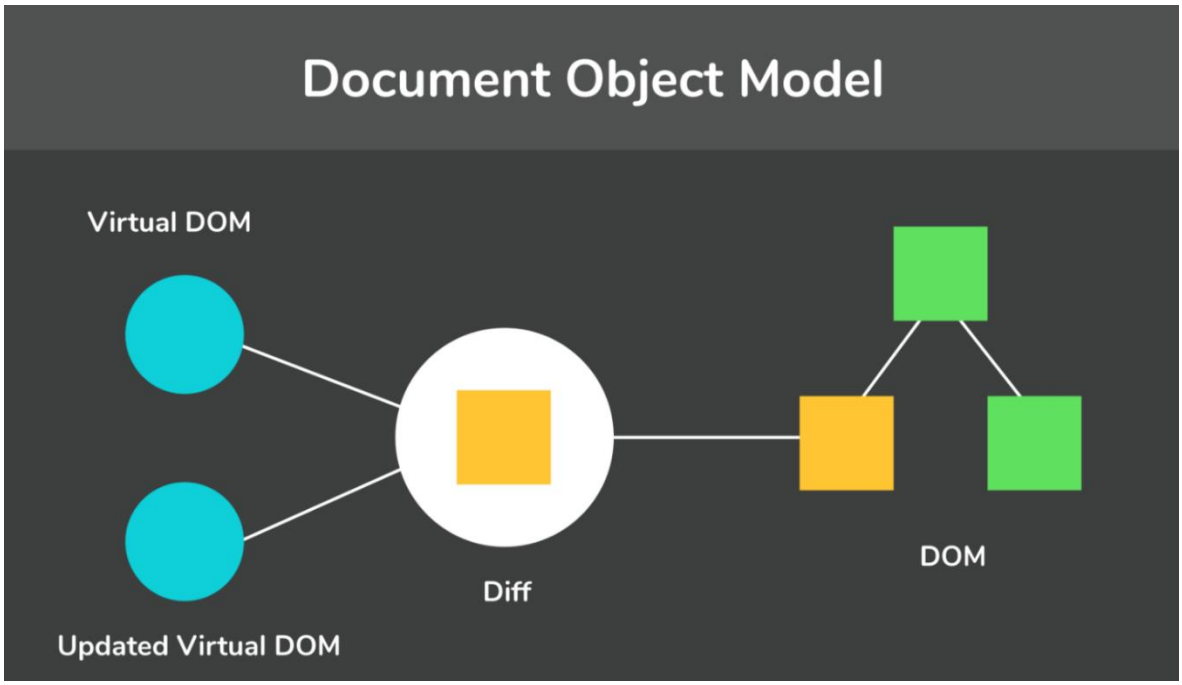


FIGURE 3. Compare and update DOM (interbit.com, 2021)

Moreover, React allows developers to use JSX as its main syntax. JSX is a syntax extension to JavaScript and is very similar to XML. While React does not require using JSX to develop React-based applications, it is prevalent among developers as it is helpful to make development easier whenever they need to write markup components and the binding events (Aggarwal, 2018). JSX helps both rendering logic and UI logic in the same components instead of separating markup and logic parts in separate files; thus, it helps developers write cleaner and more maintainable codebase for their websites (React Official Site, 2021). FIGURE 4 shows an example of how JSX helps React components become cleaner code.

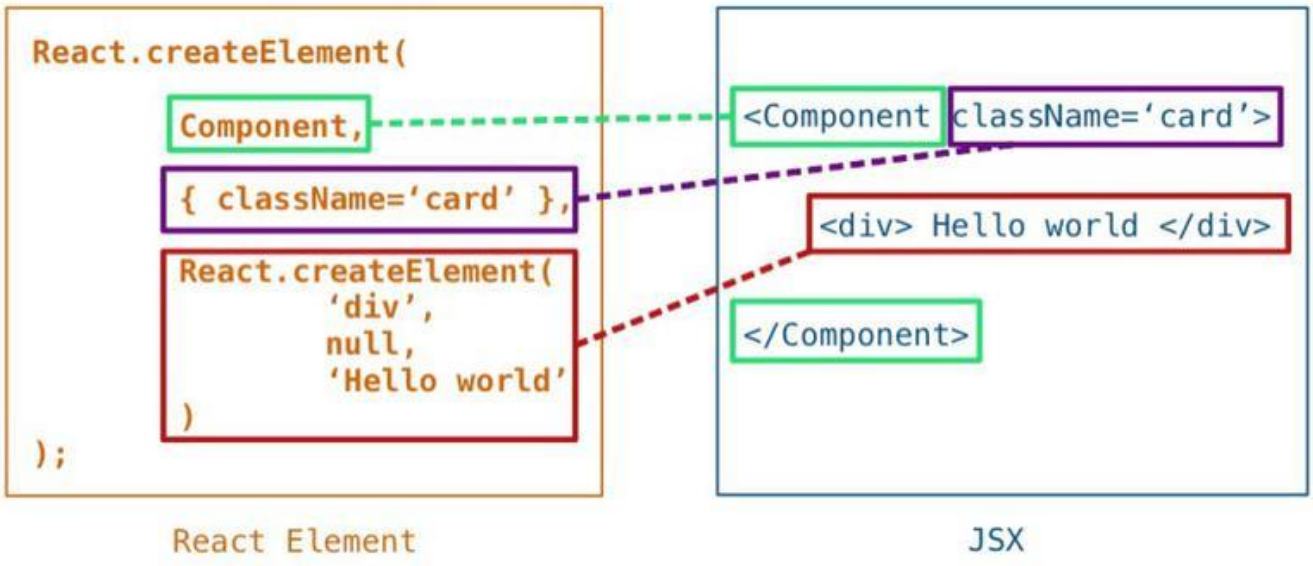


FIGURE 4. Writing React component in JSX (geeksforgeeks.org, 2021)

Furthermore, one main feature that made React has become a popular choice among developers is its reusability. Because of component-based architecture, applications built by React are straightforward to scale up, reuse as well as maintain. React built in the spirit of 'Build once, use everywhere, which helps development with this framework becomes much more manageable and understandable for new developers. That is why we can see React gaining a significant number of users over the years. Moreover, one feature that helps ReactJS stands out from other libraries is its design about data flow. React is designed to support unidirectional data flow where downstream is allowed and supported (Aggarwal, 2018). It means that React applications allow developers to nest child components within higher-order parent components. If the parent component wants to pass down its states to their child components, it can drill them via read-only properties, and then the child components can interact with their parent to update the state through call-back functions. FIGURE 5 depicts the mechanism of React's data flow.

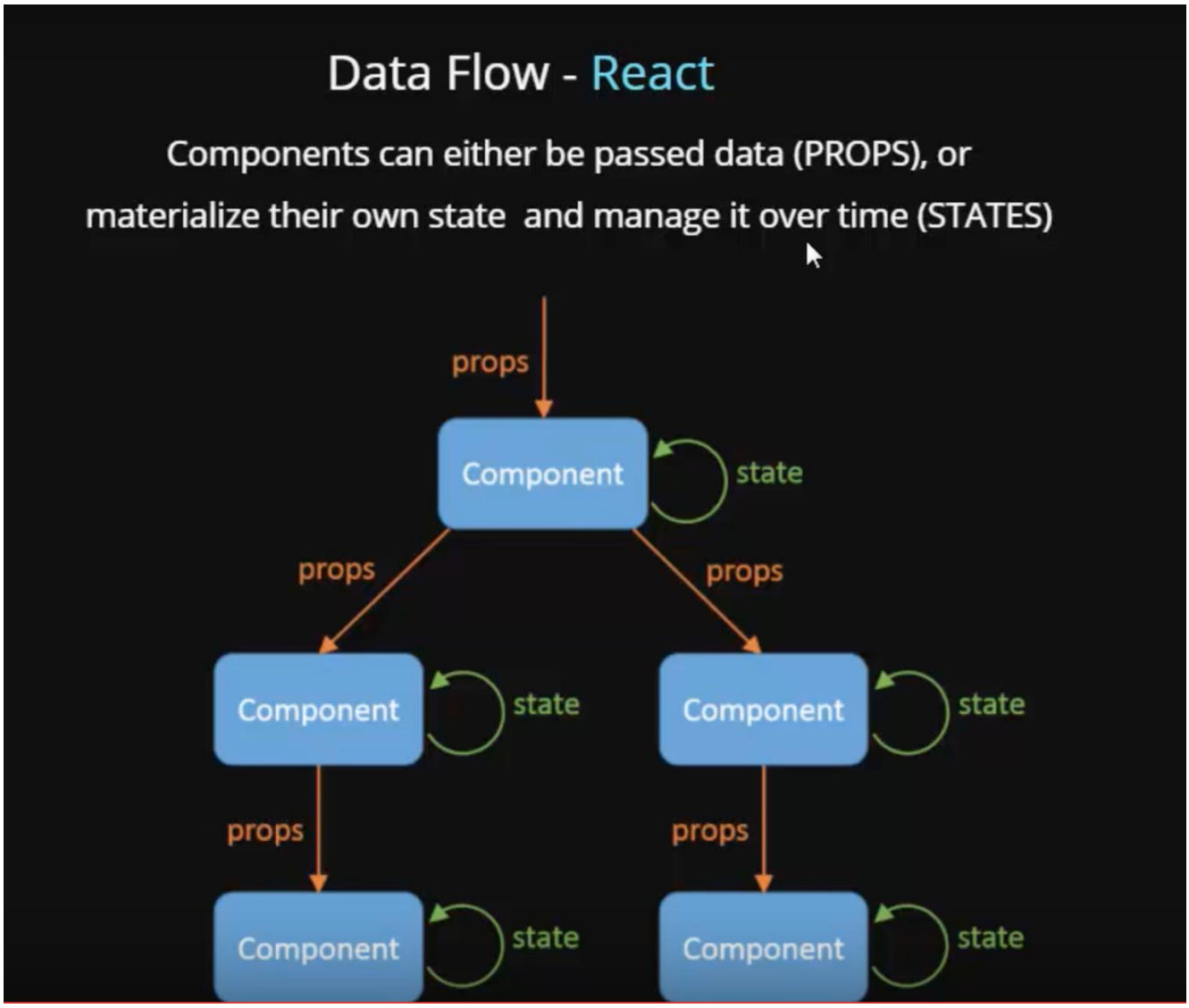


FIGURE 5. React data flow (Terai, 2017)

2.1.3 ReactJS in front-end development

As an open-source library that is utilized for building up the UIs for single-page web applications, the primary purpose of ReactJS is to provide the best possible rendering performance. ReactJS helps developers to make web applications that can use data and can change necessary parts without reloading the whole page (Aggarwal, 2018). By focusing on individual components, ReactJS empowers developers to design rich UI components. Besides, ReactJS implements a one-way data flow that is much efficient and

easier than traditional data binding (Maratkar & Adkar, 2021). With these advantages, ReactJS makes developing UI parts reliable and takes a heavyweight off from developers so they can only focus on business logic. React now has become the most favorite front-end framework and is used worldwide. In 2020, about 1,377,867 customers were using React, and this number is overgrowing, as is shown in FIGURE 6. Besides, with the release of React 18 soon, the framework will have more impact on how web development shaping. With new features such as automatic batching, new APIs, and a new streaming server renderer, React 18 can prepare multiple versions of the UI simultaneously, which will help boost the user experience to the next level (Abramov & Clark, 2021).

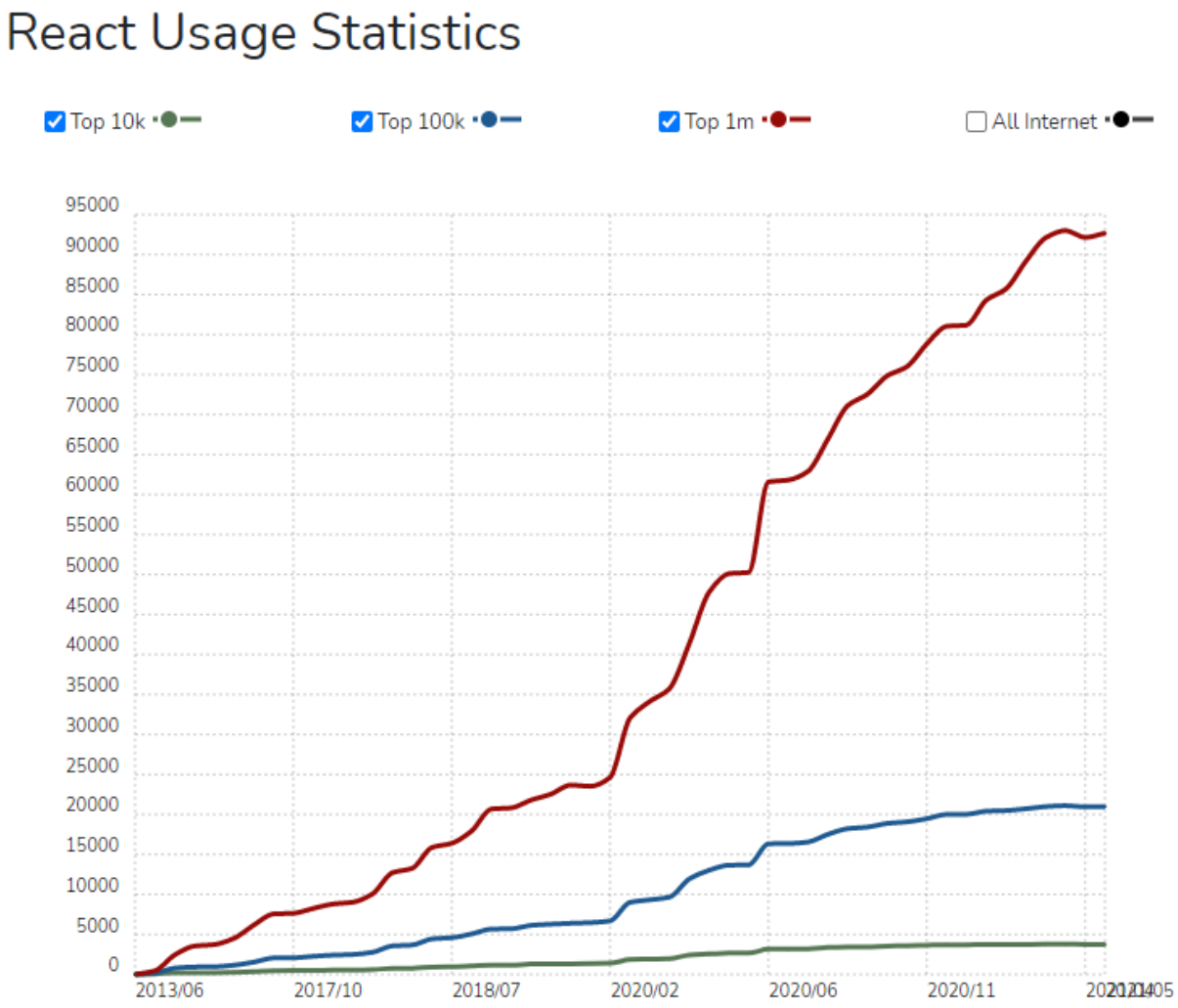


FIGURE 6. React usage statistics (builtwith.com, 2021)

2.2 GATSBYJS

After taking a general view about React and the mechanism behinds it, this section looks at GatsbyJS, a static site generator that is used as the central technology for the project, to take a closer view of its advantages as well as fallbacks to understand why it is one of the most popular choices for building a static site as this project. A static site is a site that contains fixed content, such as portfolio pages, material pages, or blogs. The advantage of static sites is that they are lighter and faster than dynamic sites because fixed content is easier to handle than dynamic content. The demand for generating static sites is increasing rapidly since nowadays people are more acquainted with the internet and have more needs for building their websites. Besides Metalsmith and Jekyll, Gatsby is the most preferred static site generator that helps people create their website with just several steps. (Imoh, 2018.)

2.2.1 GatsbyJS introduction

Gatsby is a React-based open-source and accessible framework that helps developers create blazing-fast websites and applications. Gatsby is built based on the front-end development framework React and uses Webpack and GraphQL as its core technologies; thus, Gatsby offers many advantages in performance, scalability, and security (Gatsby docs, 2021). With its excellent documentation and the latest web standards, Gatsby is one of preferable frameworks for building React-based applications. The demand for GatsbyJS in the labor market is shown in FIGURE 7.

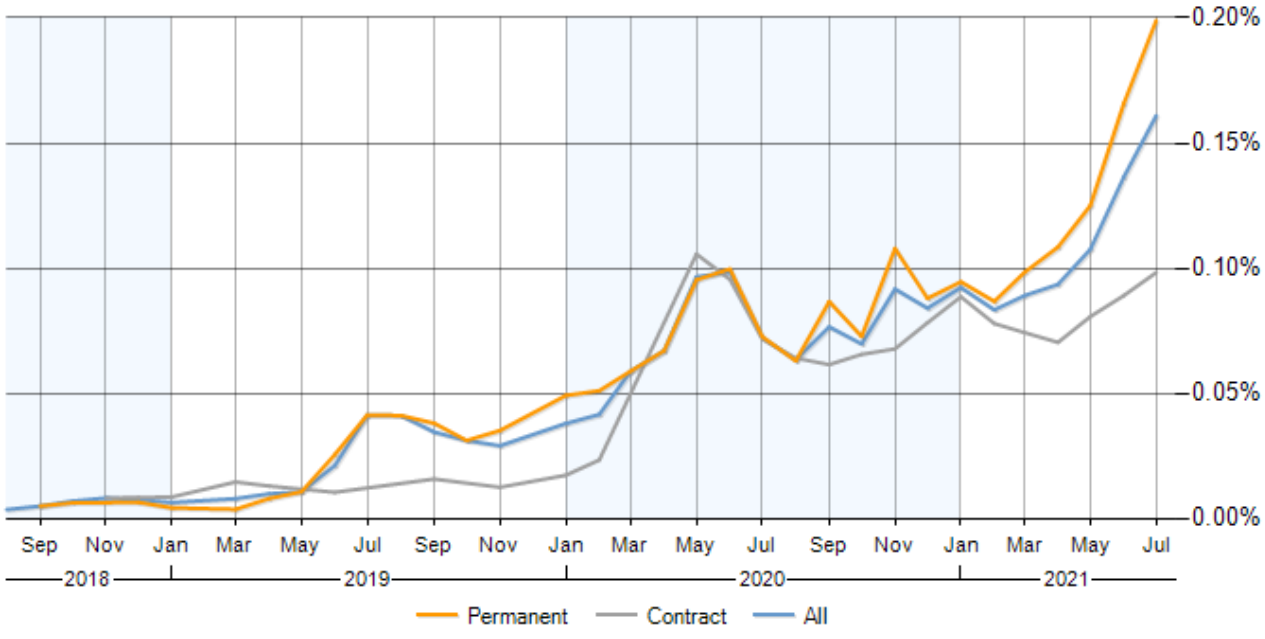


FIGURE 7. Gatsby job vacancy trend in 2021 (ITJobsWatch, 2021)

As mentioned before, Gatsby is built on top of React, which lets it inherited the strength from React. Besides, Gatsby uses a modern tech stack that is future-proof such as Webpack and GraphQL. Webpack is a JavaScript open-source module bundler that helps internally build a dependency graph that maps every module that the project needs and creates one or more bundles (Webpack Concept, 2021). By using webpack, Gatsby also takes several advantages, such as easier styling, code splitting, stable production deploys, and blazing page load speed. FIGURE 8 depicts the mechanism of Webpack in modularizing with dependencies.

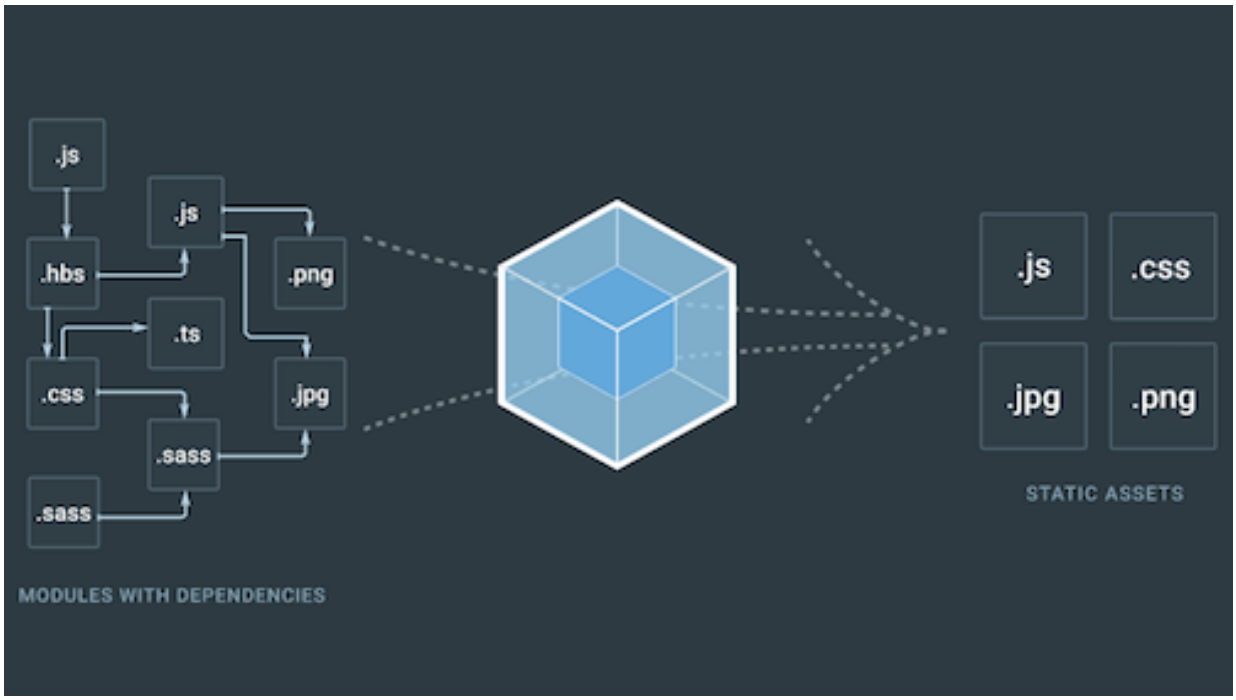


FIGURE 8. Webpack modularizes dependencies (Zimmerman, 2017)

GatsbyJS also offers built-in functionalities to use GraphQL as its primary query language for Application Programming Interfaces (APIs). With GraphQL, GatsbyJS provides a complete and understandable description of the data it requires, that helps developers to retrieve exactly what they need without having to deal with a response that includes different properties. Moreover, GraphQL queries can retrieve multiple resources in just a single request, which saves a lot of effort and time for users compares with standard RESTful API. This helps to redefine API design and client-server interaction to improve the developer experience as well as enable user experiences (Fryer, 2021). FIGURE 9 shows an example of how GraphQL makes a query.

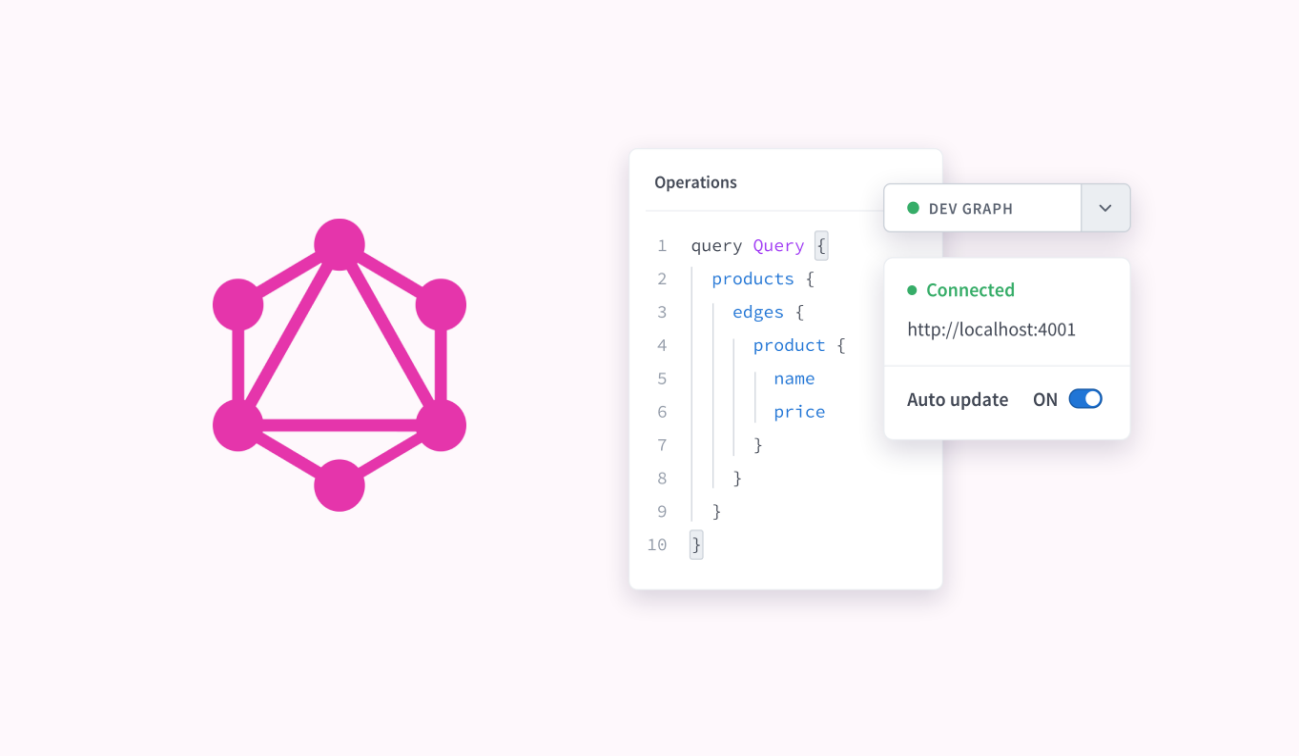


FIGURE 9. A GraphQL query example (Stemmpler, 2021)

Besides the modern technologies above, Gatsby has a great community with many plugins and starter templates. Plugins allow users to connect Gatsby with third-party platforms and import data through GraphQL queries. With these plugins, building a website with Gatsby is much easier. Gatsby does come with some default starters that help to decrease the need for boilerplate code. Moreover, Gatsby itself is an open-source framework, so its users can easily access a significant and growing number of plugins, starters, and transformers built by the existing community. (Grabski, 2021.)

2.2.2 GatsbyJS features

When talking about GatsbyJS, if there is one thing that everybody can agree on, it would be that Gatsby applications can be blazing fast. However, Gatsby comes with many other strengths, especially if developers are already familiar with React (Mischinger, 2020). The bounce rate from loading a page is relatively high nowadays. Thus, the developers have to make sure that users can measure the value that the website can bring to them. FIGURE 10 shows the relationship between the loading time and the probability of bounce.

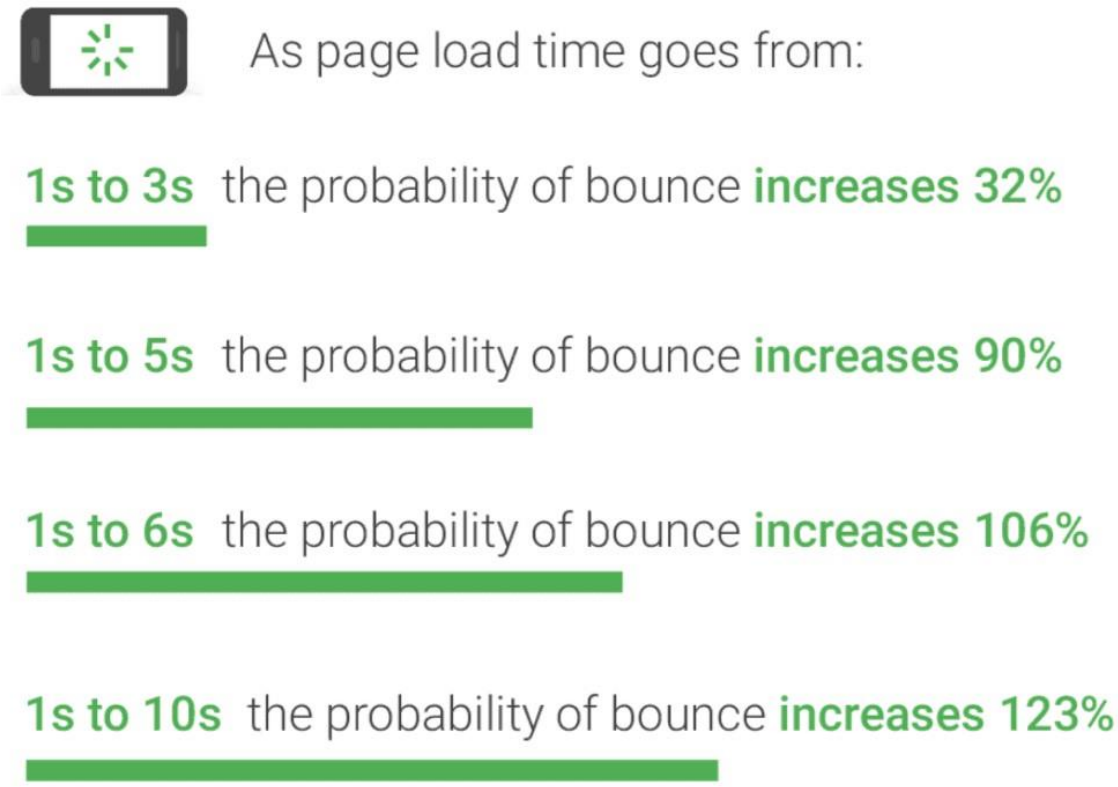


FIGURE 10. Loading time and the probability of bounce (Grabski, 2021)

According to performance tests conducted by Gatsby founder Kyle Matthews, Gatsby sites are 2-3 times faster than similar websites (Mischinger, 2020). Gatsby can increase its speed and performance because it is designed to support the server workload; thus, all the server jobs are returning a file instead of handling database queries and constructing every page as requested. After fetching all the necessary data, Gatsby will manage and handle them, and then it uses these modified data to generate static files such as HTML, JavaScript, and CSS. This pre-handling step helps Gatsby work faster in the later process. (Fryer, 2021.)

Moreover, Gatsby is supported by a big community with an extensive plugin ecosystem, robust integrations, and good documentation. As the popularity of GastbyJS is growing, and so is its community. The GatsbyJS community has already developed more than 2,000 plugins and 200,000 public Gatsby sites on Github; thus, if the developers have some problems, there is a good chance that they will be able to find an already made tool to meet their needs (Mathews, 2020). FIGURE 11 shows the number of GatsbyJS users on its own Github page.

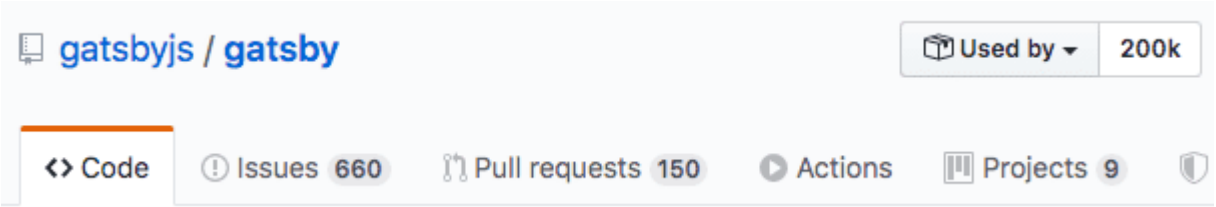


FIGURE 11. Number of Gatsby users on its Github's page (Mathews, 2020)

Moreover, one of the biggest strengths of Gatsby is its ability to load data from almost any data source (Gatsby docs, 2021). This makes Gatsby more flexible than other static site generators that limit users to only loading data from Markdown files. With Gatsby, developers can use whatever tools or backend technologies they prefer to manage application content while still using React and GraphQL on the development side. Gatsby uses source plugins to fetch data that allows developers to integrate with other third-party tools without extensive configuration, and Gatsby offers API support to add custom data as well. Each source plugin retrieves data from its source; thus, the filesystem source plugin knows how to fetch data from the file system. By including multiple source plugins, Gatsby allows developers to retrieve and combine data in only one data layer (Gatsby docs, 2021). FIGURE 12 depicts how Gatsby uses source plugins to request data with GraphQL. While developers have to make a Route for components in their pages folder when using only React, using Gatsby can directly create a component in the pages folder, and it will automatically generate a route system for that component without handling it separately.

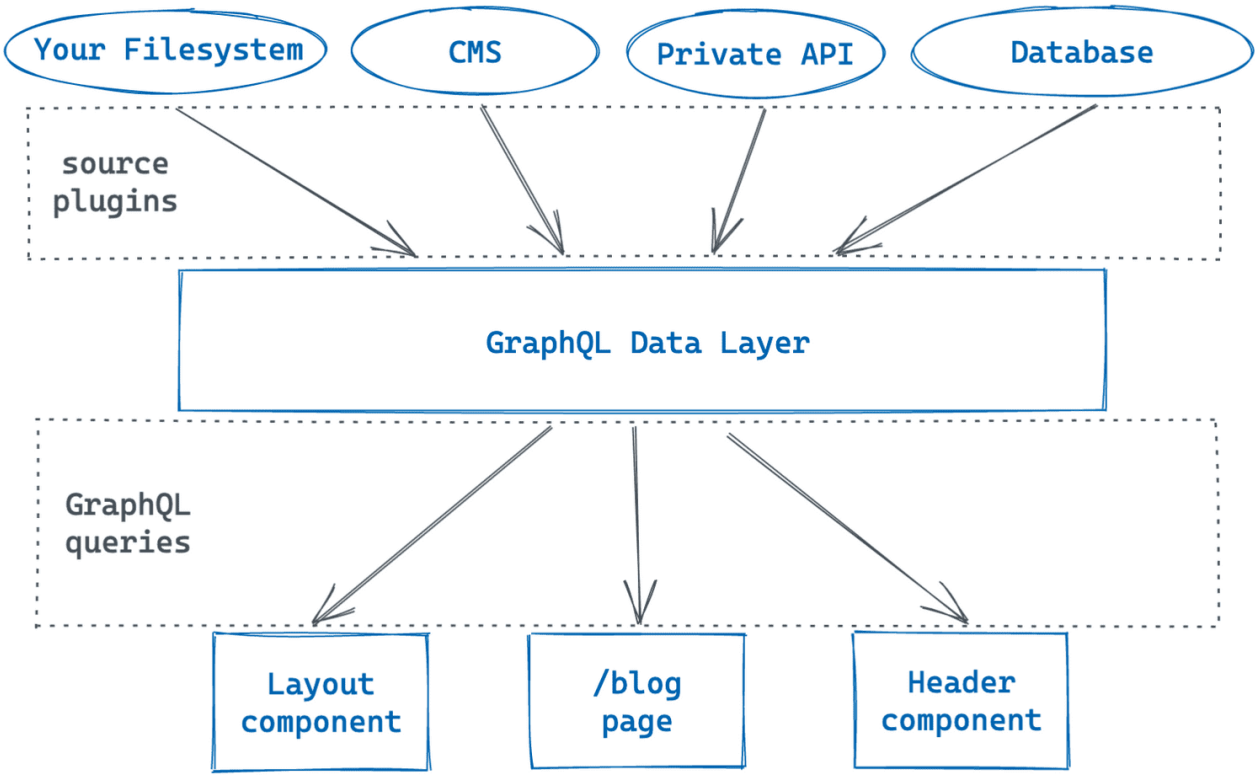


FIGURE 12. Using source plugin with Gatsby (Gatsby/docs, 2021)

Besides notable advantages, Gatsby is still evolving and has its downsides. Depending on the project requirements, there are some use cases in which GatsbyJS is not a good option. With big-scale projects, there will be much content, and the more content the page has, the longer the build time will be. Therefore, the use of Gatsby will be limited and take much time to generate a static site. Additionally, with these projects, much content updates may be required, and it makes Gatsby somehow cumbersome; thus, those updates may not be visible instantly. Furthermore, Gatsby is an open-source framework; therefore, it is challenging to have integrating standards. With many plugins and starters provided by many different sources, developers have to pay close attention to the quality of what they are getting. By using third-party tools, they have to deal with a lot of potential risks. (Grabski, 2021.)

2.2.3 GatsbyJS in front-end development

After getting general ideas about Gatsby's advantages and downsides, this section will closely look at why Gatsby is a good choice for front-end development, particularly for the template project. First, a template needs a static webpage that can serve HTML, CSS, and JavaScript and fetch the data from

Markdown files. The requirement for this is the page must be easy to create, update and modify. Therefore, Gatsby is a good choice since it can help create a static page with just some keystrokes; thus, developers can focus more on the business logic part of the page. Secondly, GatsbyJS also has the built-in functionality to fetch data from Markdown files, and it is separate from other parts; thus, it allows the users to modify the content of the template without having to handle the source code of the project. Furthermore, Gatsby offers developers complete control over the webpage's content and structures. Using Gatsby, developers can add metadata like site titles, meta descriptions, and alternative texts that help their website have more accessibility to the users, as well as help search engines to understand the content on their website to evaluate it in the search results. Besides, GatsbyJS has many advantages that help developers have better development experiences, as shown in FIGURE 13.

FIGURE 13. Pros of Gatsby for developers (Grabski, 2021)

2.3 MARKDOWN

After having a closer look at technologies that helps to build the template, besides functionalities, the project needs the data source to transfer it into the page's content. This section takes a short recap about Markdown, a markup language usually used for documentation and used as the data source for our project. Created in 2004 by John Gruber and Aaron Swartz, Markdown is a lightweight markup language that helps style a digital text document using typical formatting techniques such as headings, emphasis, links, and images (mardownguide.org, 2021). One of the notable features of Markdown is that it helps to display the document nicely in a browser is it can be optionally converted into XHTML or HTML. Nowadays, Markdown is one of the most popular markup languages globally and is widely used in writing documentation, readme files, and blogging (mardownguide.org, 2021).

2.3.1 Markdown features

There are many markup languages and formatting techniques along with Markdown. The reasons why it still is among the most popular markup languages vary from its flexibility to semantic. Firstly, Markdown can be used for almost everything. Markdown's users use it to format and document everything that needs documentation such as websites, notes, documents, books, presentations. Secondly, Markdown is portable, and it can be opened using any application. In other formatting applications which lock their content into a proprietary format, users have to use the corresponding format to open the files. Thirdly, users can write Markdown text on any device running any operating system and still read these texts at some point in the future even when the application they are using stops working. Besides, Markdown is supported by a big community. With its popularity, nowadays, you can see Markdown everywhere; thus, it is effortless to get support from its community. (mardownguide.org, 2021.)

Writing a document in Markdown is simple because it hides the logic happening under the hood, but it is necessary to look at how everything works in general. When the users write text in Markdown format, the text is stored in a plaintext file that has a .md extension, and when the browser needs to display the text file, Markdown applications will help to convert Markdown-formatted text to HTML. Markdown applications use a processor to take Markdown-formatted text then process and output it to XHTML or HTML format. (mardownguide.org, 2021.) FIGURE 14 shows how Markdown applications help convert Markdown files into HTML format so the browser can easily display them.

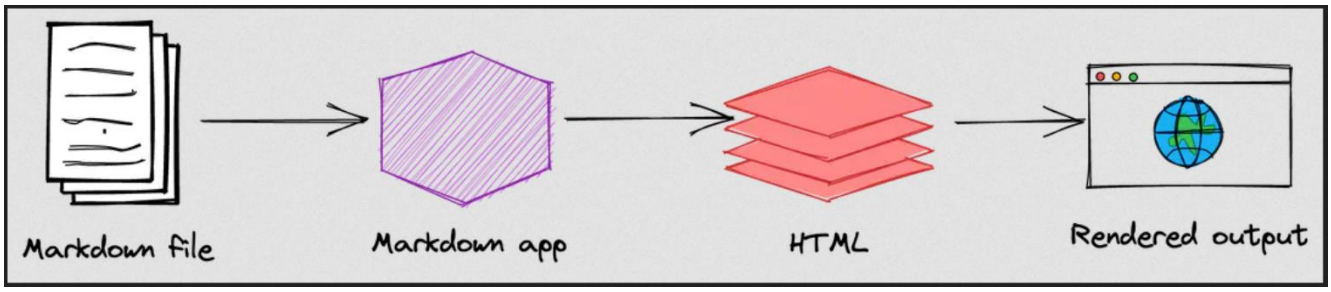


FIGURE 14. Conversion file in Markdown processor (mardownguide.org, 2021)

2.3.2 Markdown in documentation

Markdown is designed for technical documentation. There are a lot of big companies that are using Markdown for their documentation. Markdown offers a fast and simple way for creating documents and content for websites. Markdown formatting is included in line with the text; thus, it is very convenient for users when writing a document without having to stop pressing formatting buttons. With semantic and straightforward syntax, Markdown-formatted text is also easy to read in its raw state. Another reason why everybody prefers Markdown that is it is easy to get into. Its syntax is clean and clear, and once users know how to use it, they can write everything in Markdown format. Additionally, Markdown was intentionally developed for writing a website's content; thus, it comes with many web support features. Especially when it combines with Github Pages to generate a static site since it is possible to write HTML directly in any Markdown file with similar features such as headings, paragraphs, line breaks, and emphasis. (mardownguide.org, 2021.). FIGURE 15 is an example of how to write a document using Markdown syntax taken from the README.md file of this project.

```
<!-- ABOUT THE PROJECT -->

## About The Project

### Built With

- [Gatsby](https://www.gatsbyjs.com/)
- [Markdown](https://en.wikipedia.org/wiki/Markdown)
- [Github Pages](https://pages.github.com/)
- [React](https://reactjs.org/)

<!-- GETTING STARTED -->

### Installation

1. Clone the repo

```sh
git clone https://github.com/centria/template
```

2. Install NPM packages

```sh
npm install
yarn
```

3. Implement application on local server

```sh
npm start
yarn start
```
```

FIGURE 15. Using Markdown in README.md file

2.4 GITHUB PAGES

All the technologies above are about how to create and develop the project. However, to give the audiences access to the content, the process needs to host and display it. This section gives an overview of Github Pages, a static site hosting service offered by Github. Github Pages is provided by Github as a free and fast static hosting for users, organizations, and repositories that developers can customize the

domain name of their choice or just use the default domain as github.io as shown in FIGURE 16. Github Pages takes HTML, CSS, and JavaScript files from a Github repository and runs them through a process that automatically generates a static site. Therefore, Github Pages is a perfect choice for developers that only need to push generated HTML site on free hosting. There are three types of Github Pages sites: project, user, and organization. User and organization sites are linked to a particular Github account, while project sites come with their specific project hosted on Github. (doc.github.com, 2021.).

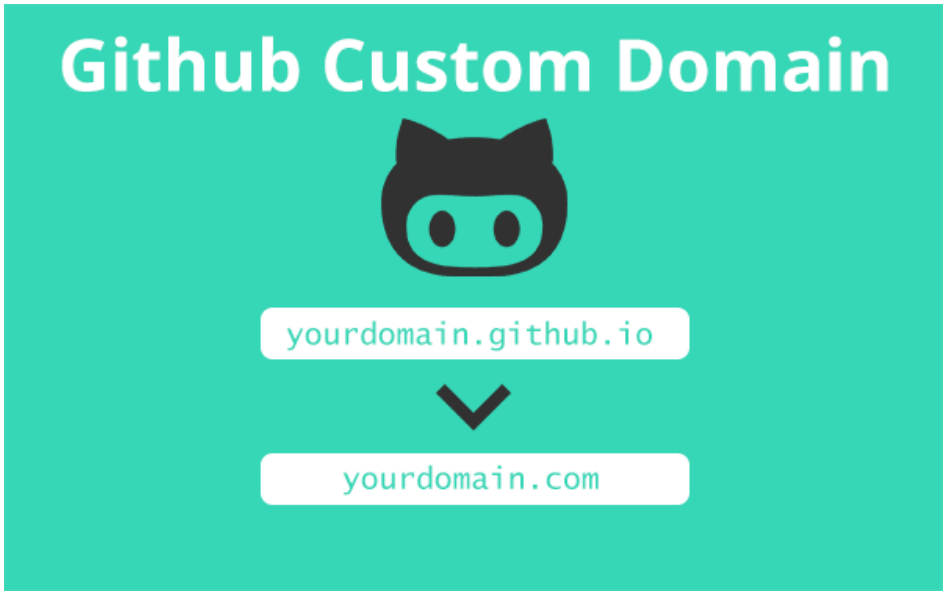


FIGURE 16. Custom domain with Github Pages (blog.webjeda.com, 2015)

Moreover, Github Pages works by looking at specific source codes of repositories on a Github account. To publish a site, Github Pages requires users to store their code in a Github repository. Furthermore, to let Github Pages features have their full effect, it is advisable to construct the source code as a typical website with a primary entry point is an index.html file. FIGURE 17 depicts the basic layout of a typical website. The publishing source for the Github Pages site is the branch and folder where the source files are stored. If the default publishing source exists in the repository, the site will be automatically generated from that source. Otherwise, Github Pages will read everything from the /docs folder to publish the site (doc.github.com, 2021).

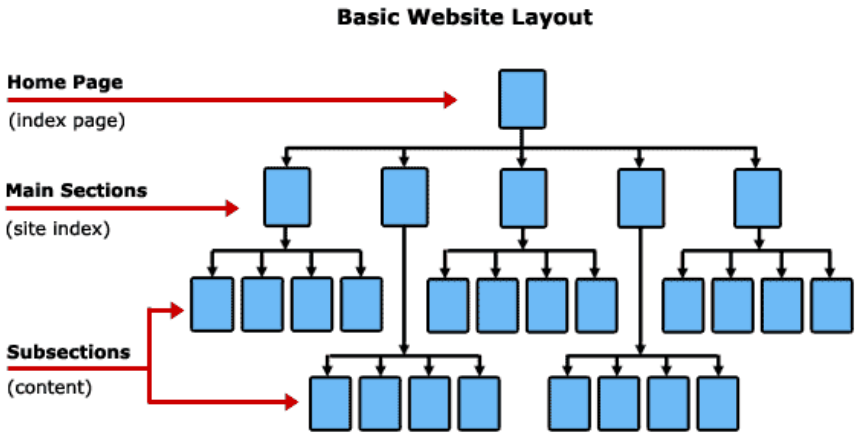


FIGURE 17. Basic website layout (Rahman, 2019)

3 THE PROJECT

The previous part takes a general view of the technologies and frameworks used in the project. Building a static website has become an essential need for everybody that uses the internet. Their needs may be a blog website to record their daily stories or a material website to give their instructions to other users. With significant use cases, the static website has become an essential part of life for everybody. This project aims to build a course material template that helps users display materials to their audiences. The following sections will depict creating the course material template with the corporation of Centria's IT department. The source code of the project is at https://github.com/centria/template , and the live demo is at https://centria.github.io/template/.

3.1 Idea and requirements

The primary purpose of the project is building a course material template for the Information Technology Department of Centria University of Applied Sciences. The main task was to design and deploy a template that could automatically populate the contents from the markdown files written before by the users. A typical software project usually has functional and non-functional requirements, as shown in FIGURE 18. To minimize problems in the development process, identifying these requirements is a crucial part of every software. Functional requirements are requirements that describe what an application or software should do. These requirements rely heavily on what type of software is being developed and what users expect (Sommerville, 2011). After the discussion with the project manager, as well as the supervisor of the project, the functional requirements for the project were defined as the project should have the functionalities to convert the page's content from data files and then display them as a static site. Besides the technical requirements, the project should meet the non-functional requirements such as being easy to deploy and maintain; the template also needs to have a modern look, clear, and clean layout so the readers can follow its contents effortlessly and the users can adjust and update their document without difficulties.

Source / Feature	Safety Regulation /Regulatory Guide	IEEE Industry Standards	Configuration Management Process Area
Function	•Specific requirement on protection system •Endorse IEEE industry Standards •Reflect to standards review plan	•Specific SCM activities •Change management •Supplier control •Tools, techniques and methodologies •Provide some samples plan	•Specific and generic goal •Specific and generic practice •Refer to other process area
Non-function	•Quality •Reliability •Quality assurance •Manage configuration activities	•Traceability •Consistency •No specific descriptions for safety system	•Integrity •Support other process area •No specific descriptions for safety system

FIGURE 18. Software requirements (Chou & Fan, 2006)

3.2 Plan

To meet the requirements and specifications of any project, planning is an important step that should be well prepared. Good preparation will save much time and effort to build a project and also make maintaining jobs more manageable in the long term. Software processes are usually related to sequences of technical, collaborative, and managerial activities. Depending on the requirements of different projects, software developers will use different software tools in their work. There are four basic process activities of specification, development, validation, and evolution in development processes. How these activities are implemented depends on different types of applications, human factors, and the structure of the organizations. (Sommerville, 2011.)

3.2.1 Gathering information

First, to have a project that can do the right things, information needs to be gathered. They can be the information about user requirements or the information about project specifications. The more understanding about the project, the better functionalities can be built. Once again, the requirements and specifications for the template need to be redefined. From the users' perspective, the template should be easy to access, and it may not require much technical or coding knowledge to use. When the users want to

deploy their documents on a new website, they just need to write new document data files and put them into a document folder. The template will help them with generating content and deployment parts. For the readers, the website's layout should have a modern look, and the contents should be easy to follow. It requires that the template is designed with a clean and clear user interface to help users have good experiences. FIGURE 19 below shows the previous course material template of Centria's IT department. At this state, its UI is not so clean and elegant and can be improved with its functionalities. Furthermore, the project should have a precise construction that can be testable and maintainable. Besides the performance efficiency, the template should be easy to extend and add more features in the future. Therefore, automation, efficiency, and easy to use were what the project aims for.

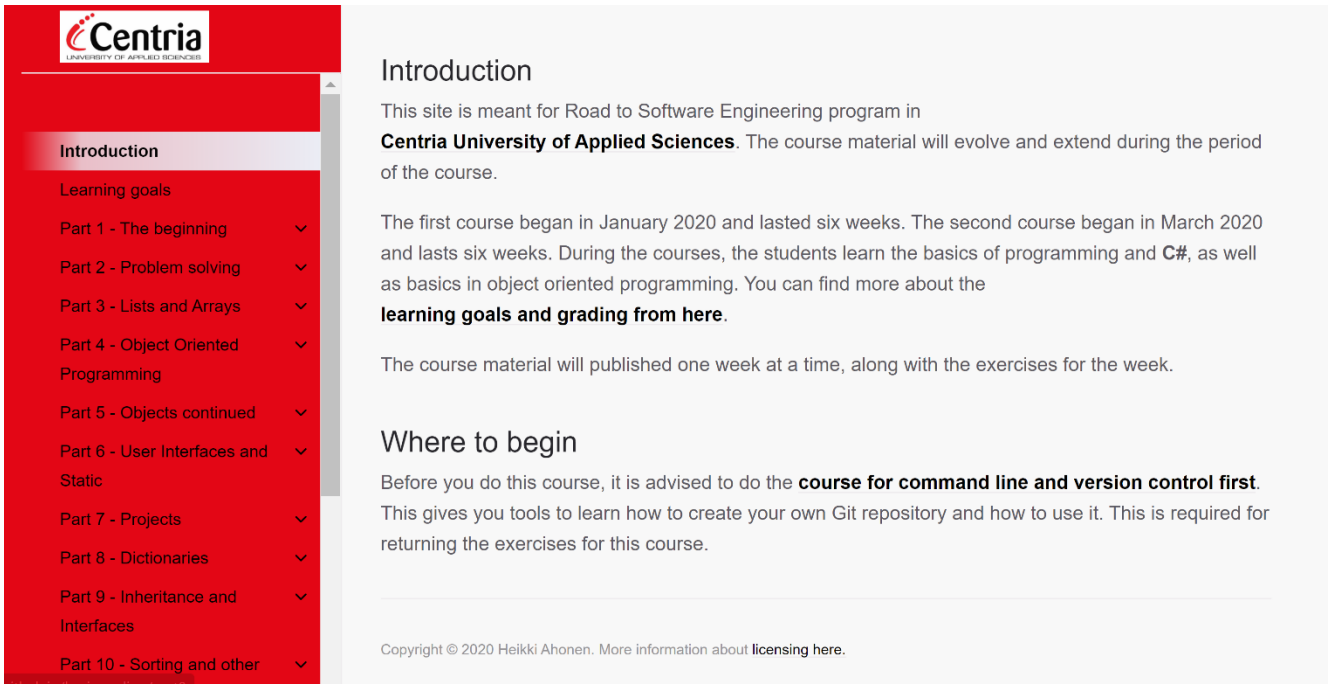


FIGURE 19. Previous course material template

After analyzing the previous course material template from Centria's IT department, all the necessary components of the project were listed to determine whether their functionality needed to be improved. The prototype of the project was sketched with main components such as the header with Centria's logo that can go back to the index page when the user clicks on it, the left navigator, which helps users navigate and commute forth and back on the website, the table of contents on the right site to navigate the content on each page that is displaying, and the footer that provides metadata about the page. FIGURE 20 shows the project sketch at the beginning stage.

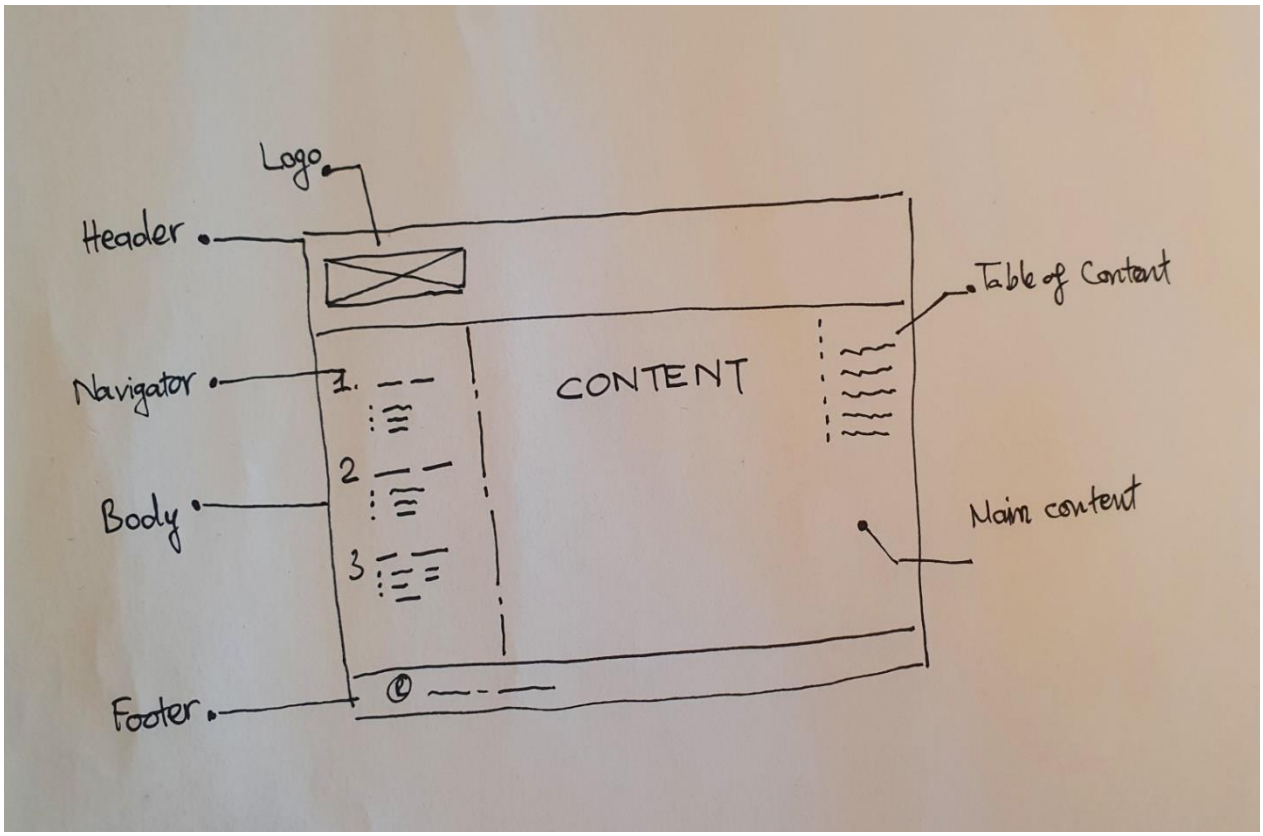


FIGURE 20. The project sketch at the beginning stage

3.2.2 Finding the solutions

After identifying the specifications, proper solutions for the project need to be defined. Since the template needed to be built a static website thus a static site generator is necessary. The Gatsby framework mentioned above is a good solution with its outstanding features. Gatsby offers an easy way to generate a static site, create its content from Markdown files, and deploy it. Therefore, the project was built bases on Gatsby and its plugins. For the documentation part, the template used Markdown files to edit the contents as it is familiar with Centria's tutors, easy to use, and is supported by Gatsby plugins. Moreover, when all the things go well, the project needs a place for hosting its content that can bring the page to the world. That is where Github Pages comes to its place. The codebase was stored on a Github server and deployed with Github Pages and Github Actions, which help the project's workflow run smoother when its code can be stored and deployed in the same place. FIGURE 21 is a demo version of the template after the development.

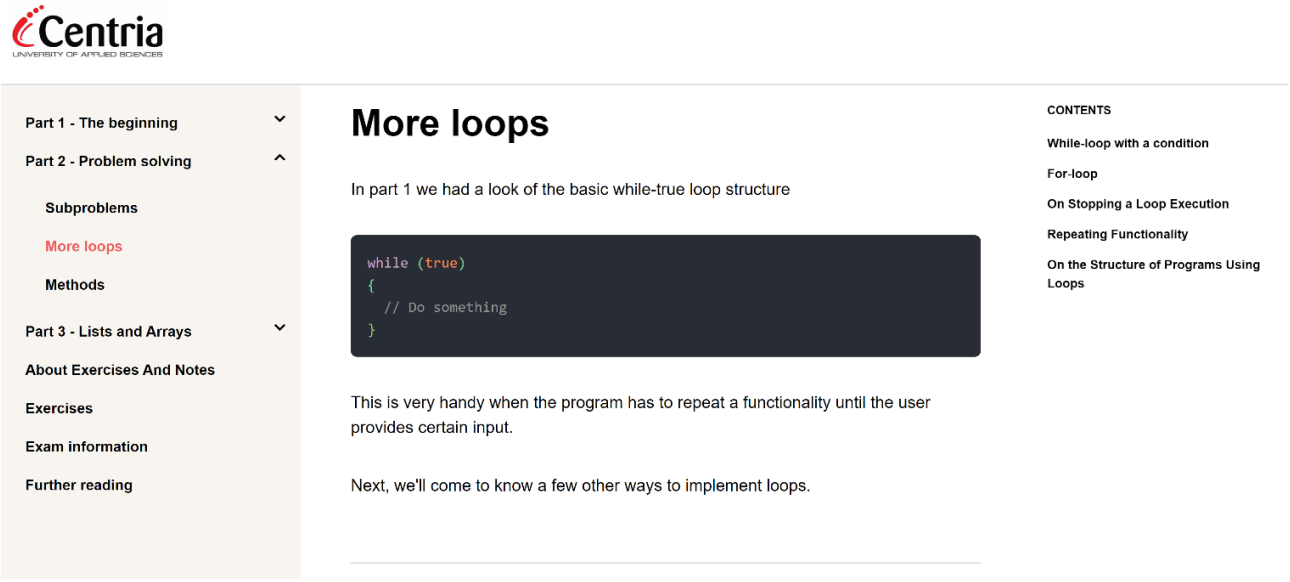


FIGURE 21. A demo version of the template

3.3 Implementation

The used Gatsby Default Starter (the default template of Gatsby) as the starting point. FIGURE 22 shows the default starter of Gatsby. After installing the default starter locally, the development environment for the project was set up. A good setup will guarantee that everything runs smoothly in the long term.

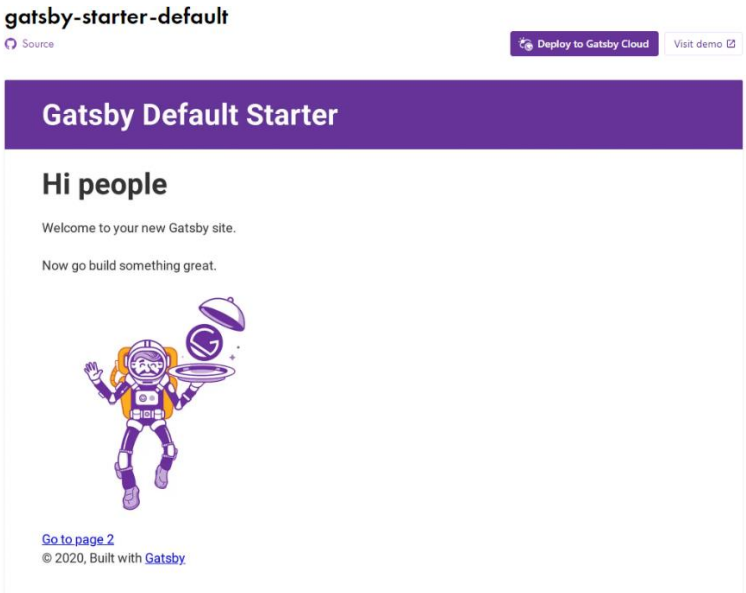


FIGURE 22. Gatsby default starter (gatsbyjs.com, 2020)

3.3.1 Setting up the development environment

First, the project needs a tool that can help to analyze and find problems in JavaScript code to make it more robust and reliable. ESlint is a good choice for JavaScript developers since it helps to find the problems with faster speed. After configuring, ESlint needs specific rules to help it understand how the source code should be analyzed. FIGURE 23 shows the specific rules that were set for the project in eslintrc file. An eslintignore file was also added to exclude node_modules, cache, and public directories for preventing ESlint from applying to these files.

```
module.exports = {
  env: {
    node: true,
    browser: true,
    es6: true,
    'jest/globals': true,
  },
  extends: ['eslint:recommended', '
plugin:react/recommended'],
  parserOptions: {
    ecmaFeatures: {
      jsx: true,
    },
    ecmaVersion: 2018,
    sourceType: 'module',
  },
  plugins: ['react', 'jest'],
  rules: {
    indent: 0,
    'linebreak-style': ['error', 'unix'],
    quotes: ['error', 'single'],
    semi: ['error', 'never'],
    eqeqeq: 'error',
    'no-trailing-spaces': 'error',
    'object-curly-spacing': ['error', 'always'],
    'arrow-spacing': ['error', { before: true, after:
true }],
    'no-console': 'error',
    'react/prop-types': 0,
  },
}
```

FIGURE 23. ESlint rules for the project

Secondly, necessary dependencies were added. Because the project mainly works with Markdown files, therefore it needs dependencies that help to handle content from those files. To transform content from other platforms and formats, Gatsby offers a plugin system that developers can find and use for specific functionalities in their applications. Two of the most critical plugins the project uses are gatsby-plugin-mdx and gatsby-source-filesystem. The first one allows the template to automatically creates pages with .mdx files in the src/pages directory and process any Gatsby nodes with Markdown media types into MDX content, while the Gatsby source plugin is used for sourcing data into the Gatsby application from the local filesystem. Lastly, to ensure the project will run smoothly during the deployment, a CI/CD (Continuous Integration and Continuous Delivery) pipeline was built to help to automate tasks like linting, testing, packaging, and deploying. By building this pipeline, the project guarantees that all the necessary setups, and tests will run before the codebase is deployed and the production can always be deployable. The pipeline that was built in the project is supported by Github Actions that is a default feature for Github's users. Since the template will be hosted by Github Pages later, using Github Actions is a convenient solution for this. FIGURE 24 shows the pipeline file for the project that can be found on the .github/workflows/main.yml file in the project.

```yaml
name: Publish on Github Pages

on:
  push:
    branches:
      - master

  pull_request:
    branches: [master]
    types: [opened, synchronize]

jobs:
  deployment_pipeline:
    runs-on: ubuntu-latest

    defaults:
      run:
        working-directory: src
```

FIGURE 24. CI/CD pipeline of the project

3.3.2 Project Implementation

After setting up the development environment, the project needs to be implemented. To separate the functionality code from the configuration code, the functionality code was put in the source directory. The main parts of the project were put in components, content, pages, and templates directories. The structure of the project is shown in FIGURE 25 below.

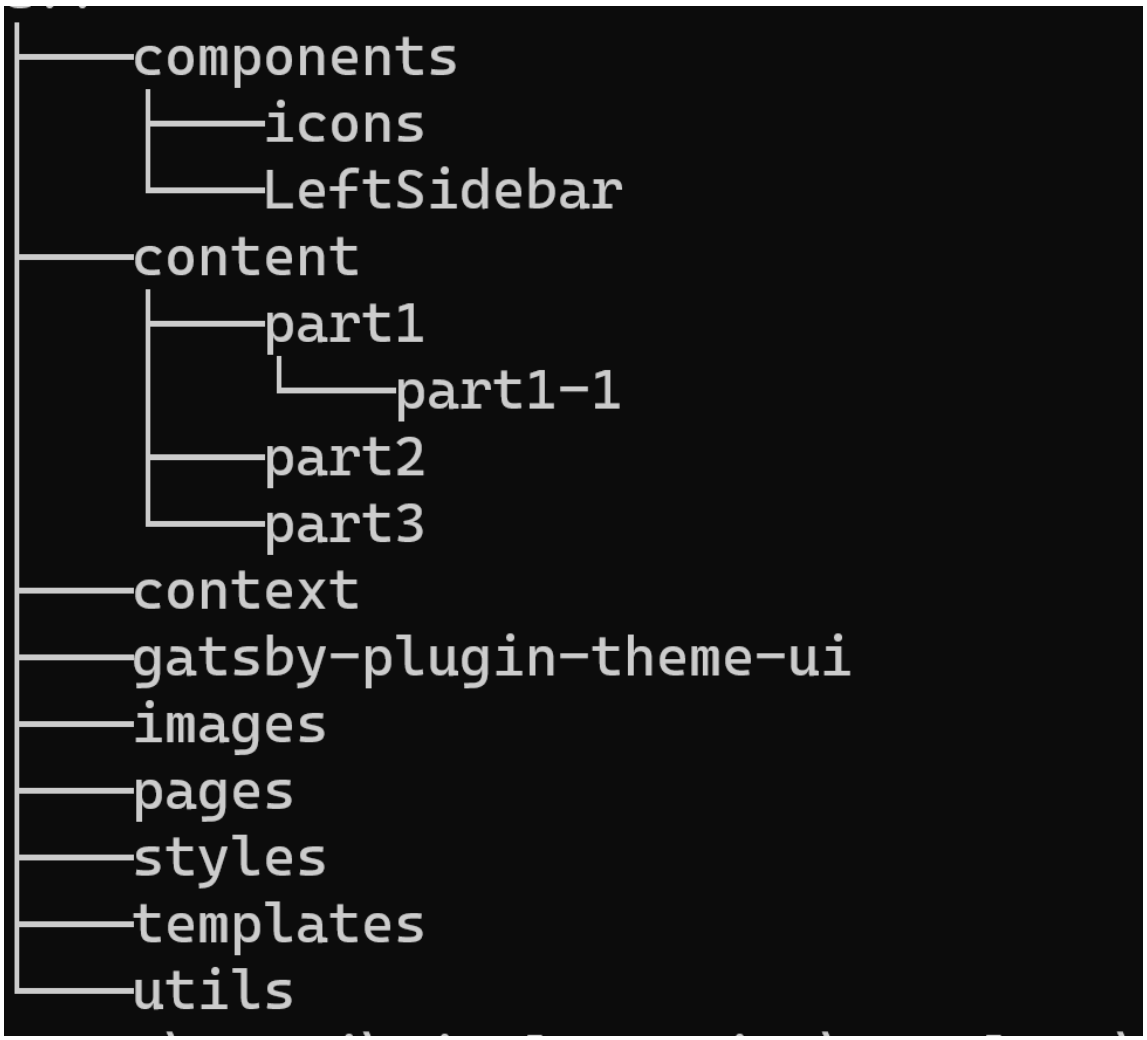


FIGURE 25. Structure layout of the project

The project is based on the ReactJS framework or GatsbyJS for more specific as its static site generator. Therefore components were used as its fundamental units. Components are independent, reusable pieces of code that are JavaScript functions. All components of the project are put in the components directory. The main components of the template are header, layout, navigator (left sidebar), table of contents (right sidebar), and item list. Each component was designed as an individual React component. Therefore, they

can be implemented, tested, and developed separately from the rest of the template. The header component is responsible for displaying the information about the document and the institution's logo that can play as a button to go back to the index page. The layout component as its name helps generate a layout for the template; thus, it can keep every page in the template synchronically. Navigator and table of contents help the audiences to navigate back and forth through the website and every single page, while the list item helps to create the main content for each page. All the source code for the components can be found in the project's components directory on its Github repository, as mentioned at the beginning.

One of essential parts of the project is the data source. While it is not responsible for the project's functionalities, it is necessary for its content creation. Therefore, it is worth investigating how data files are stored and handled in the project. The data files were stored as Markdown files in the content directory of the template. They are separated into different folders corresponding with their content. There was a folder named hidden-docs in the content directory to contain the hidden files that the developers do not want to display to the audiences directly on the website. FIGURE 26 depicts the structure of the content directory in the project.

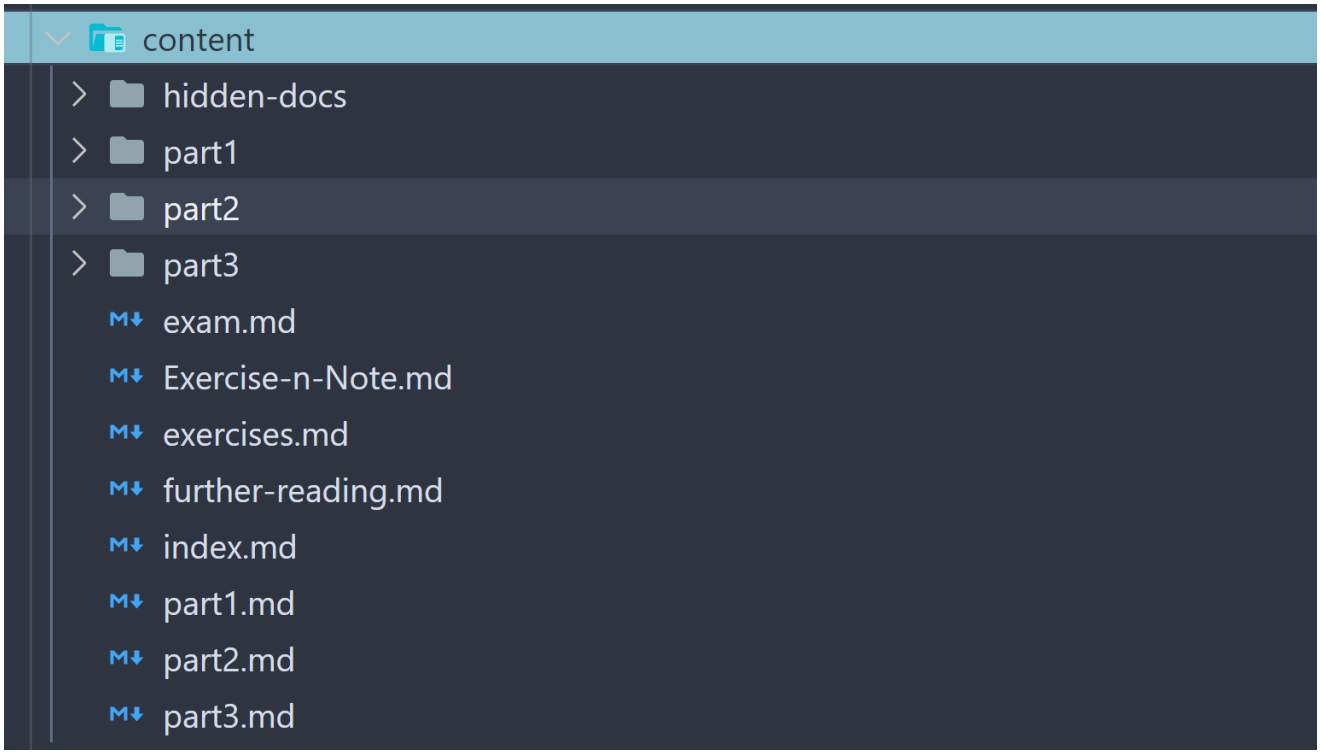


FIGURE 26. Structure of content directory

3.4 Deployment

At this stage, the template is ready to be published on a host so its audiences can access its content via a website. With the support of Github Pages, the template can store and deploy its source code in the same place. To deploy the template on a Github Pages, the users need to have a Github account created via Github's official main page as a free service. After that step, the users can go to the Github repository of the template to config their website. Firstly, they need to create a secret key on their account by going to the account setting on their Github account and generate a new personal token. FIGURE 27 shows how to create an access token on a Github account.

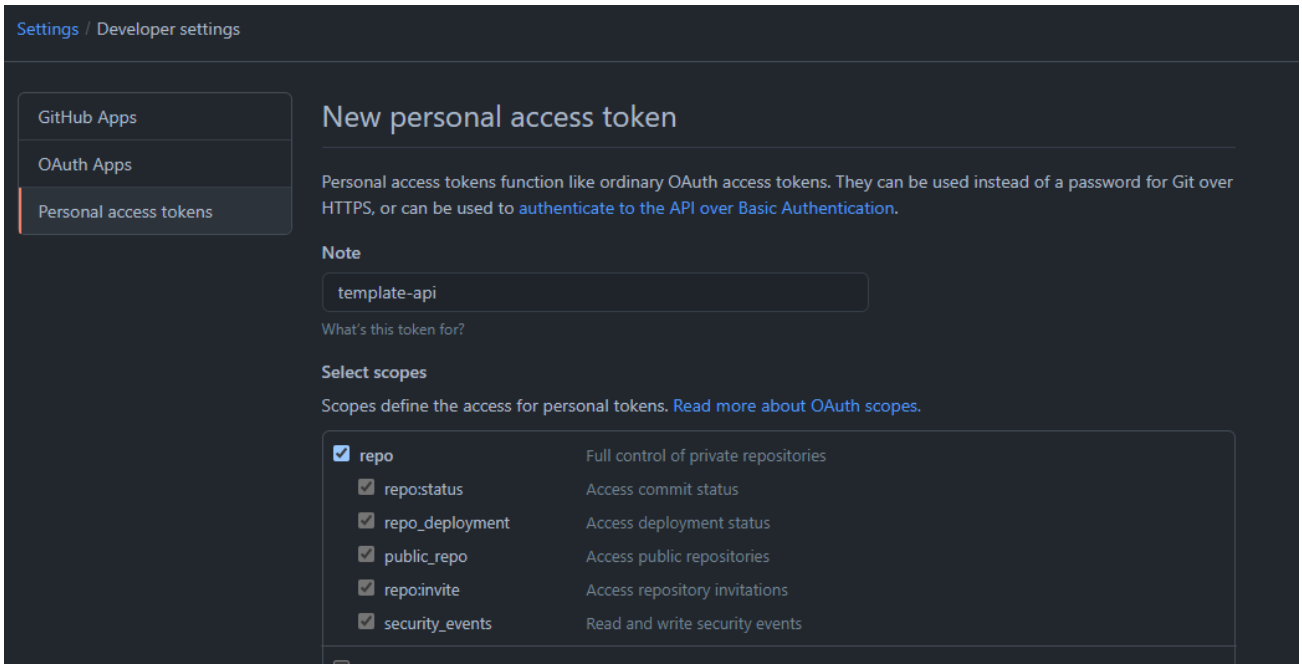


FIGURE 27. Create a new access token on Github account

When the access token is created, the users need to make sure they use the template for their new website and set it as public so other people can access it. FIGURE 28 shows how to use the template and set it as public.

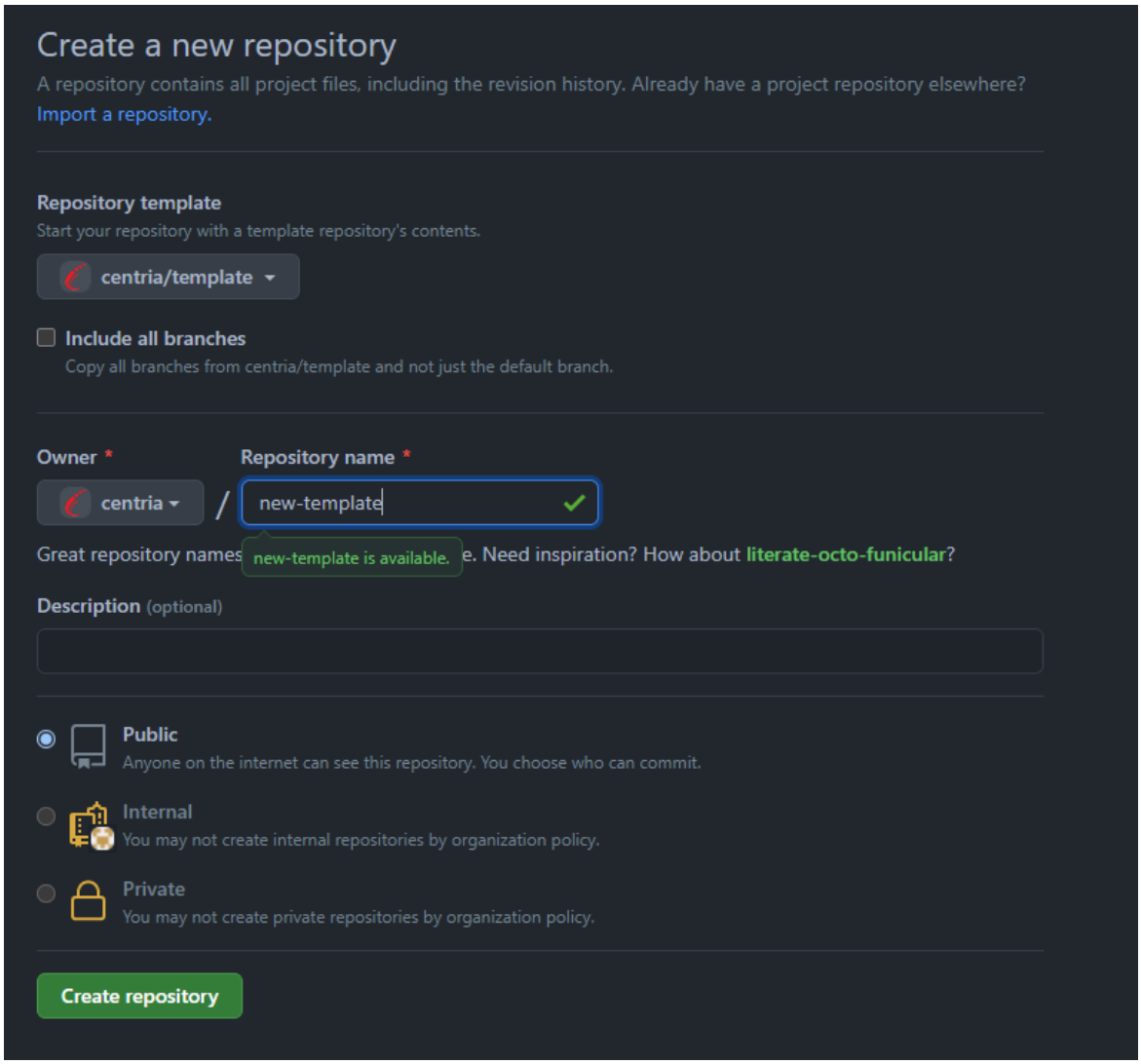


FIGURE 28. Create a new repository and set it as public using the template

After that, the newly created repository needs to set a repository secret to authenticate when an application wants to use it as its template. FIGURE 29 shows how to create a new repository secret.

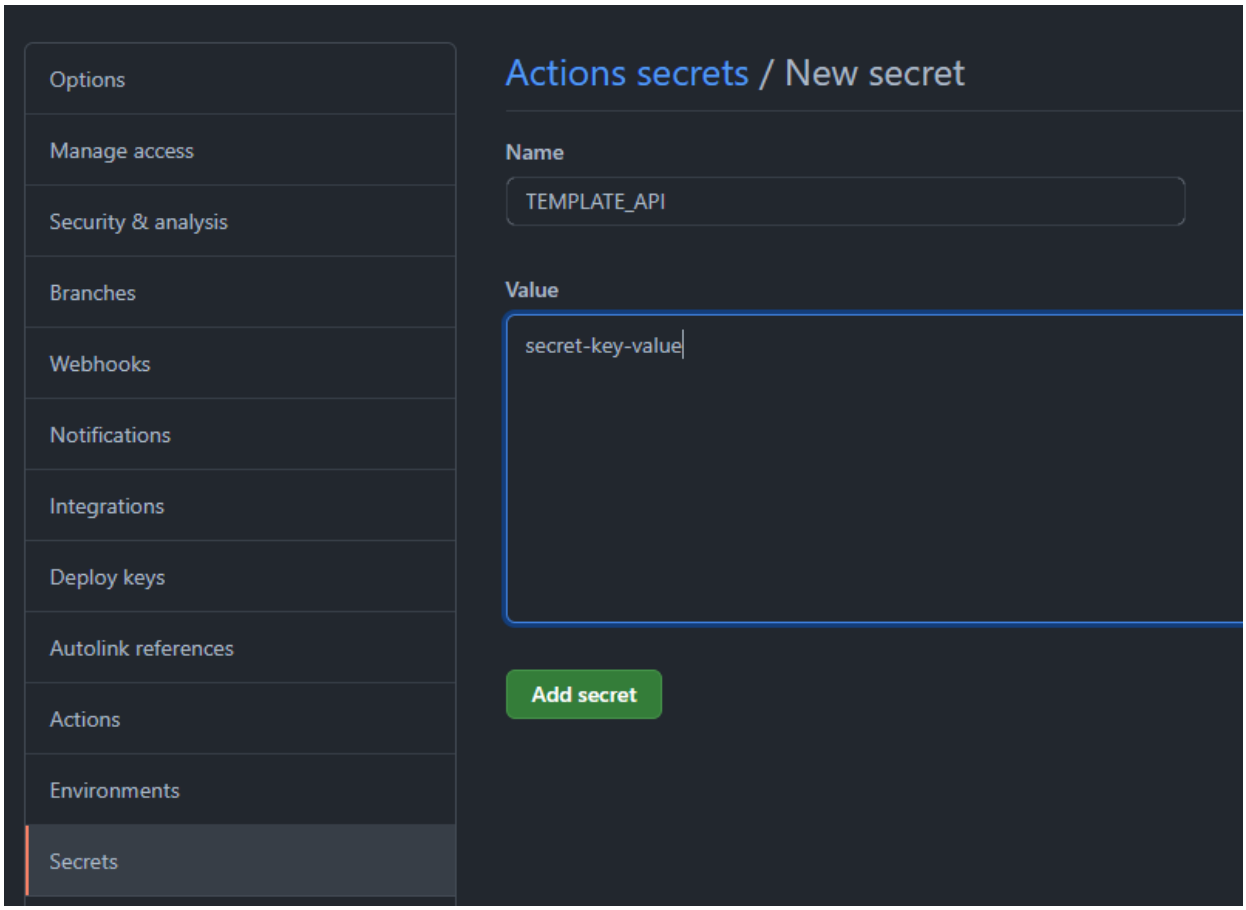


FIGURE 29. Create repository secret

After these configuration steps, the users now have a material website that contains the content as their desire. More information about the configuration for the template can be found on its Github's README.md file at https://github.com/centria/template.

To adjust the style of the page, a global styling file of the template can be found in the ./src/styles/global.js in the source code of the project. It is recommended to keep the setting as default since they will affect the responsiveness of the application on other platforms. However, users can adjust the template's styles such as color or change its responsiveness by modifying the ./src/gatsby-plugin-theme-ui/color.js and ./src/styles/media.js files. FIGURE 30 shows the default colors used in the template that users can adjust directly.

```
const colors = {
  header: '#fff', // app header's color
  text: '#000', // app text's color
  primary: '#ef5b5b', // used in hover styles
  secondary: '#c70d3a', // used in application's links
  highlight: '#efeffe', // used in highlighting background
  exercise: '#3333ee', // exercise's headers color
  sidebar: '#f8f5f1', // sidebar's color
  background: '#fff', // application background's color
  borderColor: 'rgba(0, 0, 0, 0.15)', // application border
  footer: '#ef5b5b', // footer background's color
}
```

FIGURE 30. Default colors using in the template

More information about the project's styles and how to adjust them can be found on Github's README.md file at https://github.com/centria/template.

3.5 The result

The source code of the project can be found at Centria's Github repository https://github.com/centria/template and the live demo at https://centria.github.io/template. FIGURE 31 shows the main view of the template and the home page that includes the header with Centria's logo, the main content in the center of the page, the table of contents on the top right, and the navigator on the left.

The header is also the home button that will go back to the home page when clicked. The navigator on the left displays the content of the whole website corresponding with the main heading of each page. The table of content on the top right will navigate to the content of each page, and the footer contains the information of the project's license and its copyright.

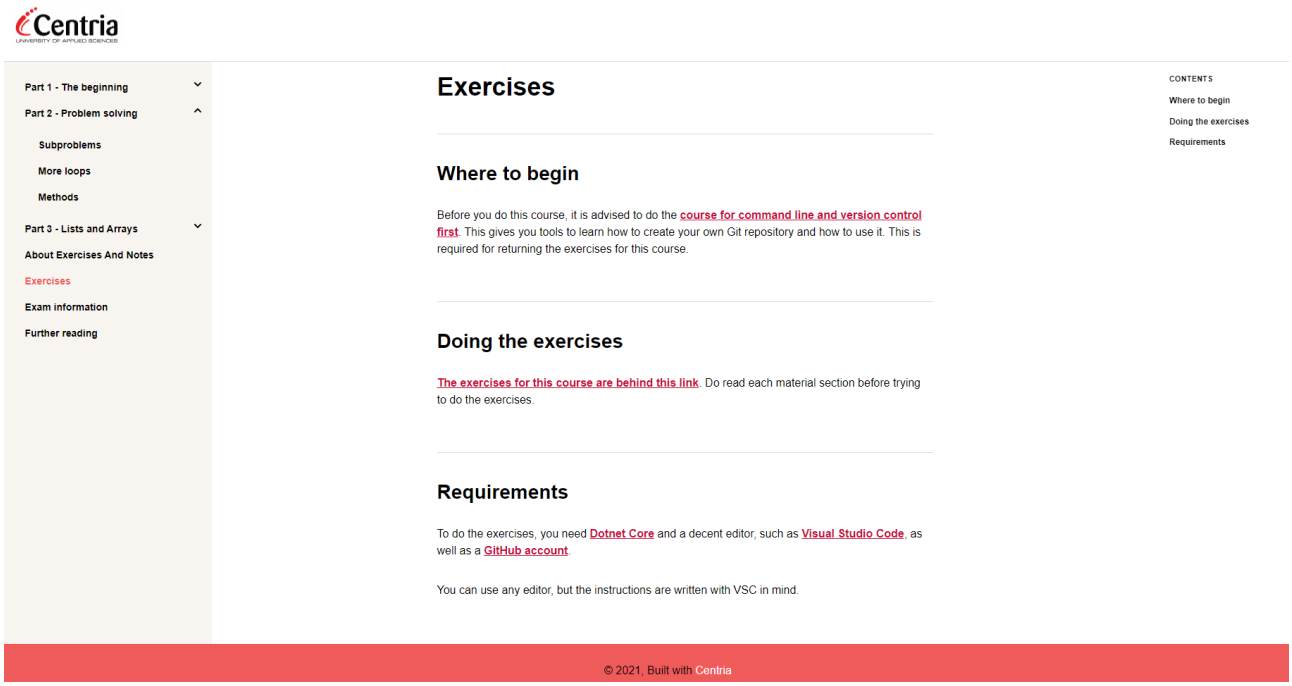


FIGURE 31. The project's home page

One of the notable features of the project is the functionality that helps the audiences with assignment instructions. This is not introduced in the previous template version. This feature helps to add extra information about an assignment or exercise or submit them to the system. For instance, FIGURE 32 shows an exercise that the lecturers want their students to do and submit to the system. The template retrieves the data of that exercise from Markdown files, then transforms and displays it on the main view of the page. Besides the content of the exercise, the students can get the submission instructions and the hints for the exercise from the corresponding parts behinds it.

Exercise 1.5

You will find following structure in the exercise:

```
using System;

namespace exercise_05
{
  class Program
  {
    public static void Main(string[] args)
      {
       string name = "Ada Lovelace";
       // Write your code here:
       }
  }
}
```

This is not a coding exorcist.

Submission Instructions

Hints

FIGURE 32. Example of an exercise in the project

4 CONCLUSION

The primary purpose of the project was to design and develop a web page template that helps its users to create and publish their websites as static sites. The new template should have a modern-looking interface and supportive features for creating and publishing websites. Since the primary goal of the template is creating a website for its users and keeping the process as simple as possible, thus the users can generate their website with minimal effort. The user interface of the template should be clean and clear. The UI components should arrange in a semantic system to help the audiences follow their content more accessible. Additionally, the template should offer basic functionalities for generating and publishing content, such as modifying the content of source files and converting them into the page's content. To meet the project's requirements, Gatsby, Markdown, and Github Pages are the most suitable technologies since their advantages can help the project not only meet the functional requirements but also be easy to use and maintain without having advanced knowledge about programming. Moreover, the assignment supporting feature is one of the notable highlights of the template since it helps lecturers add more extra information to their exercises which are not offered by the previous version. Besides, the project is well documented with instructions on Centria's Github repository, which helps its users have more accessibility to the offered features. Therefore, at this stage, the template can meet the necessary needs listed at the beginning of the project.

However, the project comes with its limitations. At this level of development, the template just offers the users basic functionalities such as content generating, modifying, and publishing. There are still many features that can be added to the project to help its audience have better user experiences. For instance, a search bar can be added on the top of the project to help the users find specific pieces of information faster, or a next button can be added at the end of each page to help the users can easier switch to the next page without using the navigation menu. Since the developing time for the project is limited, it now can only offer basic features to help create, display and publish content on a website. However, the project is designed to easily maintain and expand in the future. Thus, adding more functionalities should be implemented in a short period without much effort. Most importantly, by designing and implementing the project, the author gained many beneficial skills that can help for future development processes, such as new technologies or how to plan for software from scratch to completed application. Besides, the project also allowed the author to work and collaborate with other professional developers and learn a lot from them.

REFERENCES

Abramov, D. & Clark, A., 2021. *The Plan for React 18.*
Available at: https://reactjs.org/blog/2021/06/08/the-plan-for-react-18.html
Accessed 08 08 2021.

Aggarwal, S., 2018. Modern Web-Development using ReactJS. *International Journal of Recent Research Aspects*, 5(1), p. 133.

blog.webjeda.com, 2015. *Adding Custom Domain to Github Pages Website.*
Available at: https://blog.webjeda.com/custom-domain-github/
Accessed 28 07 2021.

builtwith.com, 2021. *builtwith.com.*
Available at: https://trends.builtwith.com/javascript/React
Accessed 16 07 2021.

Chou, I.-H. & Fan, C.-F., 2006. Regulatory Software Configuration Management System Design. In: *Regulatory Software Configuration Management System Design.* s.l.:s.n., p. 99.

doc.github.com, 2021. *About GitHub Pages.*
Available at: https://docs.github.com/en/pages/getting-started-with-github-pages/about-github-pages
Accessed 28 07 2021.

Fryer, V., 2021. *What is Gatsby JS, and How Are Ecommerce Developers Using it to Make Blazing-Fast Stores?.*
Available at: https://www.bigcommerce.com/blog/what-is-gatsby/#what-is-gatsby
Accessed 19 07 2021.

Gatsby docs, 2021. *Gatsby official documents.*
Available at: https://www.gatsbyjs.com/docs/
Accessed 19 07 2021.

Gatsby docs, 2021. *Sourcing Content and Data.*
Available at: https://www.gatsbyjs.com/docs/content-and-data/
Accessed 20 07 2021.

Gatsby/docs, 2021. *Query for Data with GraphQL.*
Available at: https://www.gatsbyjs.com/docs/tutorial/part-4/
Accessed 19 07 2021.

gatsbyjs.com, 2020. *Gatsby Default Starter.*
Available at: https://www.gatsbyjs.com/starters/gatsbyjs/gatsby-starter-default/
Accessed 30 07 2021.

geeksforgeeks.org, 2021. *React JSX in Depth.*
Available at: https://www.geeksforgeeks.org/react-jsx-in-depth/
Accessed 22 07 2021.

Grabski, T., 2021. *GATSBY JS PROS AND CONS.*
Available at: https://pagepro.co/blog/gatsby-pros-and-cons/
Accessed 19 07 2021.

Imoh, W., 2018. *Zero to Deploy: A Practical Guide to Static Sites with Gatsby.js.*
Available at: https://scotch.io/tutorials/zero-to-deploy-a-practical-guide-to-static-sites-with-gatsbyjs
Accessed 08 08 2021.

interbit.com, 2021. *React Interview Questions.*
Available at: https://www.interviewbit.com/react-interview-questions/
Accessed 22 07 2021.

ITJobsWatch, 2021. *ITJobsWatch.co.uk.*
Available at: https://www.itjobswatch.co.uk/jobs/uk/gatsbyjs.do
Accessed 19 07 2021.

Maratkar, P. S. & Adkar, P., 2021. React JS – An Emerging Frontend Javascript Library. *IRE Journals*, 4(12), p. 1.

mardownguide.org, 2021. *Getting started with Markdown.*
Available at: https://www.markdownguide.org/getting-started/
Accessed 26 07 2021.

Mathews, K., 2020. *It's Gatsby's 5th Birthday (and everyone's invited!).*
Available at: https://www.gatsbyjs.com/blog/2020-05-22-happy-fifth-bday-gatsby/
Accessed 19 07 2021.

Mischinger, S., 2020. *3 reasons why you should consider Gatsby.js for your next project.*
Available at: https://www.storyblok.com/tp/3-reasons-why-you-should-consider-gatsby-js-for-your-next-project
Accessed 19 07 2021.

Nguyen, P. A., 2021. *Arrow Hitech.*
Available at: https://www.arrowhitech.com/virtual-dom-its-definition-and-benefits-that-you-must-know/
Accessed 14 7 2021.

Rahman, M., 2019. *Fourth Chapter Lesson-4: Website Structures (Linear, Tree, Web linked, Hybrid).*
Available at: https://www.edupointbd.com/different-structures-of-websites-ev/
Accessed 28 07 2021.

React Official Site, 2021. *React Official Site.*
Available at: https://reactjs.org/
Accessed 14 07 2021.

React Official Site, 2021. *Virtual DOM and Internals.*
Available at: https://reactjs.org/docs/faq-internals.html
Accessed 14 7 2021.

Sommerville, I., 2011. Software Engineering. In: M. Horton, ed. *Requirements Engineering.* s.l.:Pearson, p. 85.

Stack Overflow, 2020. *Stack Overflow.*
Available at: https://insights.stackoverflow.com/survey/2020#technology-web-frameworks-all-respondents2
Accessed 2020.

Stemmpler, K., 2021. *What is a GraphQL query? GraphQL query examples using Apollo Explorer.*
Available at: https://www.apollographql.com/blog/graphql/examples/what-is-a-graphql-query-graphql-query-using-apollo-explorer/
Accessed 19 07 2021.

Terai, K., 2017. *Props and Data Flow in ReactJS.*
Available at: https://medium.com/@kenlynterai/data-handling-in-reactjs-c53f66b45309
Accessed 22 07 2021.

Webpack Concept, 2021. *Webpack.js.org.*
Available at: https://webpack.js.org/concepts/
Accessed 19 07 2021.

Zimmerman, J., 2017. *Webpack - A Detailed Introduction.*
Available at: https://www.smashingmagazine.com/2017/02/a-detailed-introduction-to-webpack/
Accessed 19 07 2021.