

# NEXT.js Project Generator

Group nr: 33

Kristoffer Snopestad Søderkvist

Kristoffer Eriksen Beck

## Contents

1 Introduction	<b>3</b>
1.1 Framework description	3
2 Background	<b>3</b>
2.1 Base framework	3
2.2 Similar works	5
<b>3 Method</b>	<b>6</b>
3.1 Interface Design Specification	6
3.2 Interface Design	6
3.3 Implementation	7
3.4 User testing	7
<b>4 The Design Process/Results</b>	<b>7</b>
4.1 Interface Design Specification	7
4.2 Interface Design	8
4.3 Implementation	9
4.4 User testing	10
<b>5 Resulting Framework</b>	<b>10</b>
<b>6 Discussion</b>	<b>13</b>
<b>Appendix</b>	<b>14</b>
<b>7 Code blocks</b>	<b>14</b>
7.1 Standard NEXT.js project - Template	14
7.1.1 Index.js	14
7.2 Our NEXT.js project -Template	15
7.2.1 Index.js	15
7.2.2 [pid].js	16
7.2.3 Example on a Database.json file	17
7.3 Scripts to semi automate the process	18
7.3.1 basic.sh	18
7.3.2 basic-pages.sh	18
7.3.3 basic-pages-topnav.sh	18
8 Pictures	<b>19</b>
8.1 Next.js templates	20
8.1.1 Basic-pages	20
8.1.2 Basic-pages-topnav	20

# 1 Introduction

This is a project report for Rammeverk - ITF20119, Spring 2022. The assignment of this course was to create a framework and write about the process.

## 1.1 Framework description

The framework allows the user to create different starting points when making a Next.js project, rather than the standard starting point. It also provides a stronger file structure and the option of having navigation set up, saving the user time and the ability to focus on more important parts.

# 2 Background

There are many similar solutions for generating HTML pages. But not too many that offer similar solutions as our framework. Many deliver standard Next.js by using their standard command “npx create-next-app”. It is also possible to use the command “npx create-next-app [projectName] -e [link to github template]” to use another template than their own. That is what we chose to do, not to reinvent the wheel but rather give the user more choices, and that will maybe make it a bit easier to start working with a Next.js project.

## 2.1 Base framework

### **Next.js**

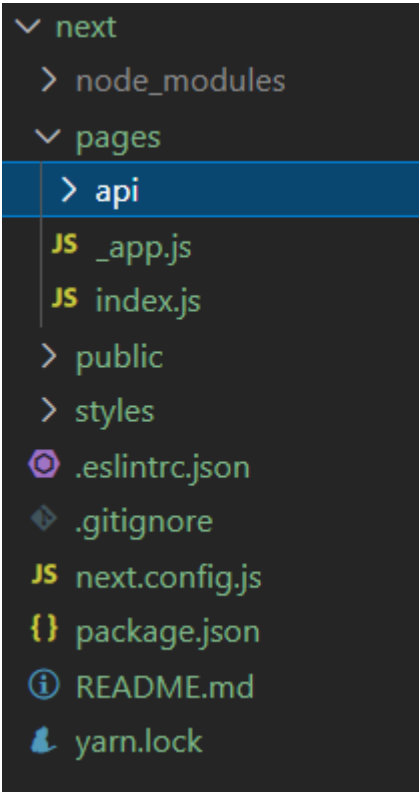
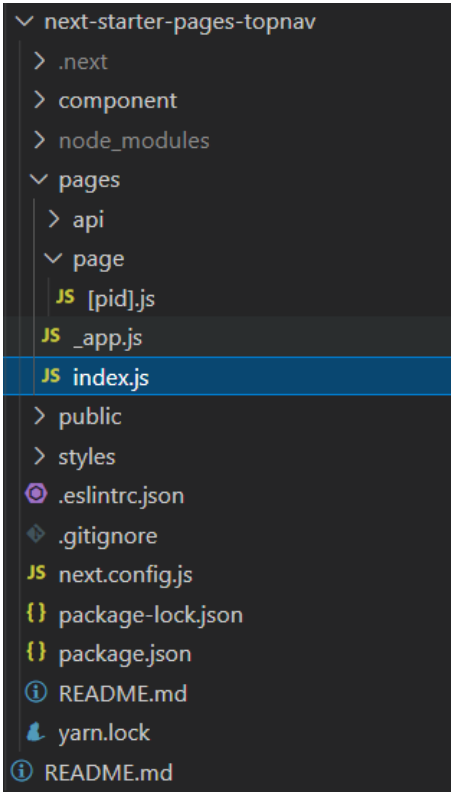
**Source:** <https://nextjs.org>

Next.js is a framework based on Node.js, which implements React. React is a JavaScript based library and focuses on web based application functionality, such as server side rendering and static websites. This has become a standard framework in

web development for those who worked with React.js and it was developed by a software engineer at Facebook. So it doesn't make sense to reinvent the wheel when it already exists.

When creating a normal Next.js project, all data is static like a normal HTML document as shown in [7.1.1 Index.js](#).

### File structure of Next.js

Standard Next.js	Our Next.js
	
<ul style="list-style-type: none"> <li>• Just what is needed.</li> <li>• No comments, need to search up what and where to change code.</li> </ul>	<ul style="list-style-type: none"> <li>• Added more structure.</li> <li>• A dynamic file. [pid].js</li> <li>• A top navigation file created in the folder component.</li> <li>• Added some comments on where the developer can change some code to make an impact on the site.</li> </ul>

## 2.2 Similar works

### **Quackit.com - HTML CODE Generator**

**Source:** [https://www.quackit.com/html/html\\_generators/html\\_code\\_generator.cfm](https://www.quackit.com/html/html_generators/html_code_generator.cfm)

Quackit is a page where the user can create a single static webpage. It gives the user a lot of input fields, and a lot of variables on each for adjusting font, text-size, spacing etc. In the end it will print the html code into a text box and the user can take that code and add it to their HTML page.

### **Webcode.tools - HTML generators**

**Source:** <https://webcode.tools/generators/html>

Webcode.tools webpage is a site design for creating each HTML element like button, tags, media, inputs etc, it can also give the user a preview of how those elements look after creating them.

### **Gohugo.io**

**Source:** <https://gohugo.io>

Just a page where the user can learn more about the HUGO framework. And learn the commands for creating a HUGO application. It is similar to commands for creating a NEXT.js application.

### **Jamstack.org**

**Source:** <https://jamstack.org/generators/>

Jamstack.org/generators page is a fast way to get to know many frameworks like NEXT.js, HUGO, Gatsby, Nuxt etc. It shows how to start coding in those frameworks or linking to some of them, and also this page can deploy a chosen project to Netlify, which is a free way to get the users page online and available.

## 3 Method

As mentioned in the introduction, this project does not use an API due to web browsers not being allowed to change things on a user's computer. The project therefore is split into two, one part creating the necessary data to create a project and the second part using that data to create a project based on the selected template.

### 3.1 Interface Design Specification

The project focuses around creating templates for the user to begin their Next.js project, so our project should then preferably focus on using Scenarios about the creation of these templates. Since we're two people on this project we are able to discuss scenarios and build as many as we need.

### 3.2 Interface Design

The design is mostly based on the templates we create, this being our main goal due to Next.js providing a standard starting point when creating a project. This also affects our scenarios and what user's could be testing

### 3.3 Implementation

Implementation requires a data source and will be created by the user, as well as the user deciding which of the templates they wish to use.

## 3.4 User testing

When testing with users our main goal is making sure each user is able to understand what they should or need to do. Main focuses being that of creating a JSON data file as well as being able to run the necessary program to launch the wanted template. Scenarios will help immensely during this as it allows us to know if something is wrong in the creation of data or reading of data, as well as seeing if we are using programs other users might not have installed, such as Next.js.

# 4 The Design Process/Results

## 4.1 Interface Design Specification

Due to our project being closer to an interface rather than an API, we have to describe our scenarios as clearly as possible to make sure the broader bases of our project are covered.

### **Scenario 1: Create a simple Next.js project.**

This scenario is to help us expand upon the basics of the Next.js starter page and allow the user to have a different starting point but still allow for the same functionality as a regular Next.js project. Using the .sh script [7.3.1 basic.sh](#). By using data similar to [7.2.3 Example on database.js file](#) we can make a simple project with multiple pages or navigation points depending on the template

### **Scenario 2: Create a Next.js project with redirects to different pages.**

This scenario will test and show ways that the created elements can have a sub-page. During creation of the dataset the user is able to add one sub-category to each input field, this will then create a pressable button with the initial text that leads to a second page filled with info from the secondary input field. Allowing the user to have multiple pages and simple navigation between these up from the start of the project rather than having to set that up themselves. Using the .sh script [7.3.2 basic-pages.sh](#)

Again, by using [7.2.3 Example on a Database.json file](#) as a reference to data, this will provide enough data to give each button and subpage the data required for a proper setup.

**Scenario 3: Create a Next.js project with a Navigation on the top.**

Some of the templates we created have a navigation bar at the top, this is similar to the previous scenario but sets up a navigation system for the user. Helping the user more easily navigate between pages rather than having to go back and forth using the browser. Not to mention the time saved from setting up these pages by hand.

Using the .sh script [7.3.3 basic-pages-topnav.sh](#)

Using data from [7.2.3 Example on a Database.json file](#), the data previously used to create subpages is now translated into making pages tied to navigation.

**Scenario 4: Create a Next.js project without a database.json file.**

This scenario is to show what happens if a user makes a Next.js project without a created JSON file. Allowing for error handling as well as showing a simple JSON setup we have as default, letting the user still utilize our template without creating a custom dataset. The user can run any of the sh scripts shown in [7.3 Scripts to semi automate the process](#) because it will run anyway, since the only step which doesn't work is the moving step.

**Scenario 5: Create a JSON file to use at other places.**

If the user wants a Json file with the same format as created in this project. The user can use the index.html page to create a json file and move it to where they want to use it. How the json file would look like using our index.html page [7.2.3 Example on a Database.json file](#).

## 4.2 Interface Design

### High-level design principles

Low barrier to entry: The use of our JSON file data generator and use of template scripts, along with a well explained “getting started” guide allow most users to easily set up a Next.js project. This does not mean that evolving and using Next.js further is easy however, understanding Next.js and the file structure might take time.



## Design patterns

Bridge: The project is divided into two parts, one part for generating JSON data, and one for reading and building based on the JSON data. This data is what combines or acts as a bridge between the two, allowing both to be used independently of one another

Decorator: While the project uses templates to build Next.js projects, the user is able to add data or wanted fields to the creation of the project. Letting the user somewhat control how their project looks in the beginning as well as having navigation settled if they so desire it.

## 4.3 Implementation

The project started out with wanting to offer the user a simpler or improved way of creating a Next.js project. The first goal was for us to achieve a low barrier to entry for the user, even though the setups want additional data in the form of JSON. The project contains default values whenever a project is created and still gives the user access to navigation and subpage setups depending on the template, even without providing or creating a dataset. Due to unforeseen problems in the form of web browsers not allowing direct changes to a user's computer, the project had to be split into two halves. One part provides a dataset in JSON based on user input as well as letting the user see examples of the templates we provide, the other part being the creation of the Next.js project based on a template. Depending on the data provided and the selected template, the user gets to customize their starting point within limits but can have navigation and subpages set up rather than having to build it from the ground up.

## 4.4 User testing

Due to very limited time, we were only able to do minor user testing. We were able to get a friend to test out our framework using our "getting started" guide. He pointed out some missing and vague points in the guide, as well as pointing out the template examples on the data generator page were placed weirdly. There wasn't much in terms of revising the interface except making some semantic changes and updating the "getting started" guide. We also looked into other potential programs that might

affect the building of the project, one potential of this is a program called “Git-for-Windows” as that is what the script runs through. None of us know if this is a standard on most Windows PCs or if it comes from downloading Git related software

## 5 Resulting Framework

This page is created for making the process of creating a Next.js project easier. If the user wants some data already on the web page created with NEXT.js.

The screenshot shows the 'Next.js - Generator' web application. It features a dark purple header with the title 'Next.js - Generator'. Below the header is an 'Installation guide' section with five steps: Step 1 (Node.js download), Step 2 (template selection), Step 3 (database.json download), Step 4 (script execution), and Step 5 (localhost launch). A 'Submit' button is located below the guide. Below the submit button is a 'New Input' section with two text input fields: 'Webpage title:' and 'Title :'. Below these fields is an 'Add more content' button. Below the button is a section titled 'Select script based on image templates' with a 'basic.sh' option. The main content area is a white box with the title 'This is a title', a subtitle 'Get started by editing pages/index.js', and a recommendation 'Recommended to use Netlify to host this webpage'. At the bottom of the white box are four buttons labeled 'Number 1', 'Number 2', 'Number 2', and 'Number 4'.

In the file “scripting.js” will take the data from this index.html page and convert it over to a JSON.file, this file is going to be used to add data to the chosen project in Next.js. The JavaScript snippet below is how we execute the conversion.

```
var obj = { title: webpageTitle, content:[] };
```

```
let title = allInputSection[index].querySelector('input').value;
let data =
allInputSection[index].querySelector("input.subInput").value
let id = allInputSection[index].querySelector('input').id

    obj.content.push({
      "text": title,
      "data": data,
      "id": id
    })
  }
}
var json = JSON.stringify(obj)
```

The result after running the [7.3.1 basic.sh](#) script with a database.json file. It will launch a server, and the start webpage will look like this.

# This is a title

Get started by editing `pages/index.js`

Recommended to use Netlify to host this webpage

Number 1

Number 2

Number 2

Number 4

More templates and how those look like are added at the [8 Pictures](#) chapter. Those were the result after everything to create a Next.js project is done.

This framework is made to make it easier for a developer to start using Next.js for their web page project. And if they already have some content to display they just need to fill the index.html page with that content and run the script.

Right now it does not support a json file with a name other than “database.json” or the code will just use the json file if it has the data with a specific format. So data management is a bit limited but if the developer wants to change it from using the json in the code to some other database or hardcoding, it is easy to do so.

## 6 Discussion

### **What went well?**

To create a project based on our template in Next.js, this was an important step in our project since otherwise it would just be a normal Next.js framework.

Also went well to create a .sh script to create a next project, and also move the “database.json” file to its location.

### **What could have worked better?**

The automation process could have worked better in this project, but as we found out the hard way almost all web browsers will not allow the web page to automatically run scripts on its computer. So now the user/developer needs to run the scripts manually, but this is not a problem for most people when following the “Getting Started” guide.

The database could have worked better, now it just imports the “database.json” to the web page. Originally the plan was to add the json file to the project and it should be possible to use it as an online database when it was running. The hope was to use it as an API with GET, POST requests.

## Appendix

### 7 Code blocks

This section is to show a lot of code blocks and to not clutter up all the text above. It will show off the files from a normal Next.js Framework project, and some of our files. The code snippets are all the code in the files, but just the important part.

#### 7.1 Standard NEXT.js project - Template

##### 7.1.1 Index.js

```
export default function Home() {  
  return (  
    <div className={styles.container}>  
      <Head>  
        <title>Create Next App</title>  
        <meta name="description" content="Generated by create next app" />  
        <link rel="icon" href="/favicon.ico" />  
      </div>  
    )  
  }  
}
```

```
</Head>

<main className={styles.main}>
  <h1 className={styles.title}>
    Welcome to <a href="https://nextjs.org">Next.js!</a>
  </h1>

  <p className={styles.description}>
    Get started by editing{' '}
    <code className={styles.code}>pages/index.js</code>
  </p>

  <div className={styles.grid}>
    { /* In a standard start for NEXT.js is they are showing a hardcoded way to
display data.  */ }

    <a href="https://nextjs.org/docs" className={styles.card}>
      <h2>Documentation &rarr;</h2>
      <p>Find in-depth information about Next.js features and API.</p>
    </a>

    <a href="https://nextjs.org/learn" className={styles.card}>
      <h2>Learn &rarr;</h2>
      <p>Learn about Next.js in an interactive course with quizzes!</p>
    </a>

    <a
      href="https://github.com/vercel/next.js/tree/canary/examples"
      className={styles.card}
    >
      <h2>Examples &rarr;</h2>
      <p>Discover and deploy boilerplate example Next.js projects.</p>
    </a>
```

## 7.2 Our NEXT.js project -Template

### 7.2.1 Index.js

```
//Importing Json here
import Data from "../api/data/database.json"

export default function Home() {
  //Adding Json to var

  // Change Data[0].title to change
  var title = Data[0].title;
  // this is an array with strings. So either change it out with another
  array.
  // Or add it manually in the return statement below.
  var arrString = Data[0].content;

  //-----
  return (
    <div className={styles.container}>
      <Head>
        {/* The title is used here */}
        <title>{title}</title>
        <meta name="description" content="Generated by create next
app"/>

        {/* Change icon in the folder public*/}
        <link rel="icon" href="/favicon.ico"/>
      </Head>

      <main className={styles.main}>
        {/* The title is used here */}
        <h1 className={styles.title}>
          {title}
        </h1>

        <div className={styles.grid}>
          {/* Array of strings is used here. This is a type of foreach
code. Creating a more dynamic page and showing how it could be done in an easy
way. */}
```

```

    {arrString.map(key =>
      <a href={"/page/"+key.id} className={styles.card}>
        <p>{key.text}</p>
      </a>
    )}
  </div>
</main>

```

### 7.2.2 [pid].js

This file is a dynamic file. Which is in use every time /page/pageId is called upon.( /page/1 )

```

const DynamicPage = () => {
  // This is a dynamic page. Every url that is after /page/[here] will
  start this file.

  const router = useRouter()
  // Getting url data.
  const pid = router.query
  // Converting url data to title, string data and an ID.
  const title = pid.text;
  const data = pid.data;
  const id = pid.id;

  // Below it is the HTML where title, data and id is used. Change them
  out with a string below or above
  return (
    <>
      <header>
        <title>{title}</title>
      </header>
      <main className={styles.main}>
        <section className={singleStyles.sectionMid}>
          <section className={singleStyles.returnFrontpage}>

```



```

        <Link href="/" passHref>
          <h1>To the frontpage</h1>
        </Link>
      </section>
      <h1>{title}</h1>
      <p>Id : {id}</p>
      <p>{data}</p>
    </section>
  </main>
</>
)
}

```

### 7.2.3 Example on a Database.json file

```

[{"title":"Webpage title","content":[
{"text":"Article 1","data":"Content to article 1","id":"1"},
{"text":"Article 2","data":"Content to Article 2","id":"2"},
{"text":"Article 3 without a subfield","data":"","id":"3"}
]]

```

## 7.3 Scripts to semi automate the process

### 7.3.1 basic.sh

```

# Run one of the next.js versions created.
npx create-next-app basic -e
https://github.com/Bass4Nation/NEXT-JS-Generator/tree/main/next-starter
# Move the created json to next js project
mv database.json ./basic/pages/api/data/
# Go into folder created
cd basic
# Start server
npm run dev

echo("To close the terminal press CTRL + C")
# Terminal won't close itself until keypress CTRL + C
read -r -d '' _ </dev/tty

```

### 7.3.2 basic-pages.sh

```
# Run one of the next.js versions created.
npx create-next-app basic-pages -e
https://github.com/Bass4Nation/NEXT-JS-Generator/tree/main/next-starter-
pages
# Move the created json to next js project
mv database.json ./basic-pages/pages/api/data/
# Go into folder created
cd basic-pages
# Start server
npm run dev

# Terminal won't close itself until keypress CTRL + C
read -r -d '' _ </dev/tty
```

### 7.3.3 basic-pages-topnav.sh

```
# Run one of the next.js versions created.
npx create-next-app basic-pages-topnav -e
https://github.com/Bass4Nation/NEXT-JS-Generator/tree/main/next-starter-
pages-topnav
# Move the created json to next js project
mv database.json ./basic-pages-topnav/pages/api/data/
# Go into folder created
cd basic-pages-topnav
# Start server
npm run dev

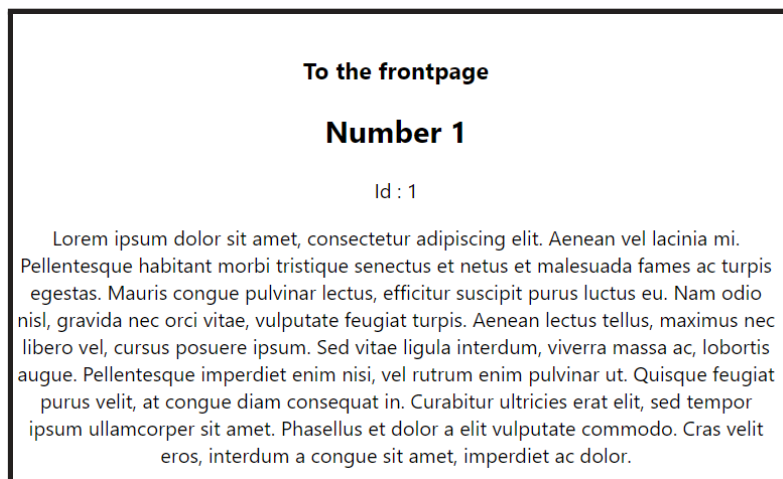
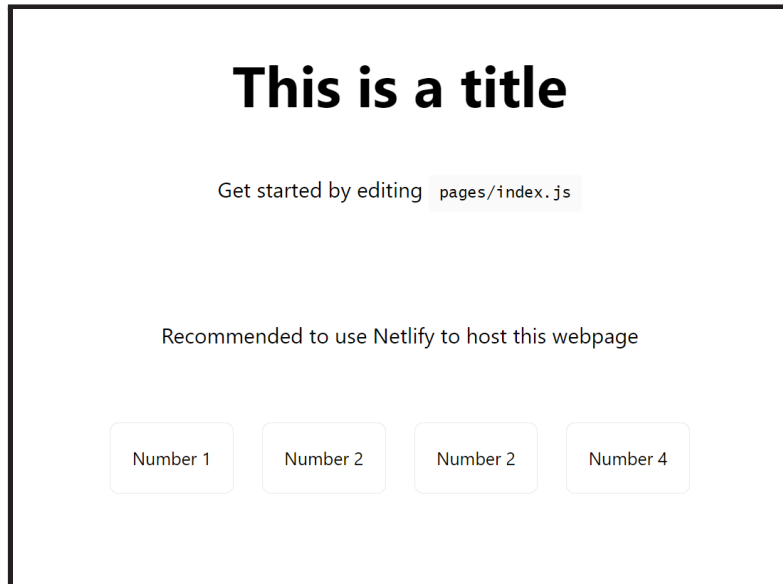
# Terminal won't close itself until keypress CTRL + C
read -r -d '' _ </dev/tty
```

## 8 Pictures

This chapter is just to show off some pictures of our project. How some of the templates would look like when the developer has chosen what to start with.

## 8.1 Next.js templates

### 8.1.1 Basic-pages



### 8.1.2 Basic-pages-topnav

