Tutorial - KickOff für deine Landingpage mit Svelte

Hey moin,

schön, dass du in mein Tutorial schaust. Ich habe dieses Tutorial erstellt, weil ich genau wie du wahrscheinlich auch ganz frisch im Programmieren mit Snel und Svelte bin und den Schmerz fühle, wenn man Stundenlang an einem Fehler rätselt, um zu merken, dass man einfach eine Variable in die falsche Komponente gepackt hat.

Ich hoffe ich kann dir mit meinem Tutorial einige Kopfschmerzen sparen und dir eine gute Basis bieten mit, der du durchstarten kannst. Viel Spaß!

Den Code zu meinem Tutorial findest du unter folgendem GitRepo:
https://github.com/Cliff99999/Graphischer-Editor-f-r-GitHub-Pipelines---Landingpage

Aufbau aud Projekt aufsetzen

Ich habe mich dazu entschieden die Landingpage in 6 unterschiedlichen Komponenten zusammenzufassen, welche in der App.svelte „App Komponent“ zusammengefasst sind:

- Banner
- Service
- About
- Testimonial
- Social Media
- Footer

Zuerst muss ein neues Projekt aufgesetzt werden. Nutze dazu folgenden Befehl in deinem Terminal:

```
npx degit sveltejs/template “Projektname”
cd digital_agency_using_svelte
npm install
```

Falls du noch kein Deno installiert hast, dann kein Stress. Auch diese Kopfschmerzen entnehme ich dir gerne. Check einfach das Deno Install Tutorial in meinem Gitrepo ab. Ich habe dort jeden einzelnen Schritt mit Screenshots hinterlegt. Ist Idiotensicher.

CDN einfügen

In diesem Projekt arbeiten wir mit einem sogenannten CDN (Content Delivery Network). Füge hierzu einfach folgenden Coe in den „index.html“ Ordner ein:

```
<link                                          rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/b
ootstrap.min.css"
```

```
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw
1T"crossorigin="anonymous">

<!--fontawesome.js-->
<script                                                    defer
src="https://use.fontawesome.com/releases/v5.0.6/js/all.js"></
script>

<!--Slim.js-->
<script  src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6ji
zo"
crossorigin="anonymous"></script>

<!--bootstrap.js-->
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/boo
tstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07j
RM"
        crossorigin="anonymous"></script>
```

Styles

Zusätzlich kannst du meinen folgenden CSS-Code nutzen um deinem Code einen Style zuzufügen. Füge ihn in den Ordner „global.css“ ein. Du findest ihn im public Ordner:

```
:root {
  --gradient: linear-gradient(90deg, #0072ff 0%, #00d4ff 100%);
  --light: #fff;
  --grey: #f8f9fa;
}

html,
body {
  margin: 0;
  padding: 0;
  font-family: Arial, Helvetica, sans-serif;
  scroll-behavior: smooth;
}
.main-bgcolor {
  background-image: var(--gradient);
}
.light-color {
  color: var(--light) !important;
}
.grey-bgcolor {
  background: var(--grey);
}
.company_brand {
```

```
  font-size: x-large;
  font-family: cursive;
}

.section {
  padding: 50px 0;
}

.section-body {
  padding: 20px 0;
}
```

Static Data

Du hast nun eine erste Basis mit der du unkompliziert loslegen kannst. Ich werde dir nun zusätzlich noch meine sogenannte „Static Data“ zur Verfügung stellen. Du kannst mithilfe dessen so gut wie alle Änderungen, die du in Texten auf der Landingpage machen willst, direkt hierüber machen. Erstelle einfach in deinem src. Ordner einen neuen Ordner namens „Data“ und erstelle hier drin nun eine neue „File“ namens „data.js“. Füge dort nun folgenden Code ein:

```
const HEADER = "Graphisher GitHub Actions Editor";

const NAVBAR_DATA = [
  { id: 1, url: "/", label: "Home" },
  { id: 2, url: "#services", label: "Services" },
  { id: 3, url: "#about-us", label: "About us" },
  { id: 4, url: "#testimonials", label: "Testimonials" },
  { id: 5, url: "#footer", label: "Contacts" }
];
const BANNER_DATA = {
  HEADING: "Kein Stress, kein Code und nur Erfolge! Wir
revolutionieren GitHub Actions",
  DECRIPTION:
    "Bearbeite jetzt deinen GitHub Actions-Workflow mit  unserem
graphischen GitHub Actions Editor!",
  TUTORIAL_URL:
    "https://www.thinkwithgoogle.com/intl/en-gb/marketing-
resources/programmatic/google-digital-academy/",
  WATCH_TUTORIAL: "Zum Tutorial"
};
const SERVICE_DATA = {
  HEADING: "Worum geht es in unserem Projekt?",
  ALL_SERVICES: "KEIN STRESS, KEIN CODE UND NUR ERFOLGE! WIR
REVOLUTIONIEREN GITHUB ACTIONS",
  SERVICE_LIST: [
    {
      LABEL: "Mehr aus GitHub rausholen",
      DESCRIPTION:
        "Die     meisten     Entwickler     assoziieren     die
Quellcodeverwaltung mit GitHub. Die Plattform, kann  jedoch für
viel  mehr  als  nur  das  Synchronisieren  von  Git-Repositories
```

```
verwendet werden. Unter anderem kann die Workflow-Engine GitHub
Actions dafür eingesetzt  werden fast alle Ereignisse innerhalb
GitHHub zu automatiseren. Sie bietet somit eine großartige
Lösung für Continuous Integration/Continuous Deployment (CI/CD)-
Pipelines.",
      URL: "images/service8.png"
    },
    {
      LABEL: "Was sind überhaupt CI/CD- Pipelines?",
      DESCRIPTION:
        "Du kannst einen CD-Workflow so konfigurieren, dass er
ausgeführt wird, wenn ein GitHub-Ereignis auftritt z. B. wenn
neuer Code in den Standardzweig deines Repositorys übertragen
wird. Dies geht nach einem festgelegten Zeitplan, manuell oder
wenn ein externes Ereignis auftritt. Diese Befehle müssen
normalerweise extra gecoded werden und genau hier kommen wir
ins Spiel!",
      URL: "images/service4.png",
      ID: "service-img"
    },
    {
      LABEL: "Wir wir dir das Leben erleichtern",
      DESCRIPTION:
        "Unser graphischer Editor für GitHub Actions gibt dir
die volle Kontrolle. Mit dessen Hilfe kannst du per Knopfdruck
Workflows ausführen, die den Code in deinem Repository erstellen
und auch Tests darüber ausführen. Konfiguriere deinen CI-
Workflow beispielsweise so, dass er ausgeführt wird, wenn ein
GitHub-Ereignis auftritt, z. B. wenn neuer Code in dein
Repository übertragen wird. Wir ermöglichen dir sogar einen
festgelegten Zeitplan einzustellen oder feste Ereignis zu
definieren. Das alles ganz ohne Code!",
      URL: "images/service9.png"
    }
  ]
};

const ABOUT_DATA = {
  HEADING: "Warum unseren Editor nutzen?",
  TITLE: "Fakt  ist",
  IMAGE_URL: "images/network.png",
  WHY_CHOOSE_US_LIST: [
    "Wir sind der einzige GitHub Actions Editor auf dem Markt",
    "Wir ersparen  dir Zeit und Nerven.",
    "Wir haben alle Funktionen implimentiert, die du brauchst.",
    "Wir sind schnell",
    "Unser Kundenservice ist 24/7 telefonisch erreichbar.",
    "Wir lieben was wir tun.",
    "Du kannst Teil von etwas großem werden.",
    "Wir arbeiten für deinen Erfolg"
  ]
};
const TESTIMONIAL_DATA = {
  HEADING: "Was unsere Nutzer sagen",
```

```
  TESTIMONIAL_LIST: [
    {
      DESCRIPTION:
        "Der Grafische  GitHub Actions Editor hat mit seiner
einfachen  Handhabung einen großen Unterschied für meine Arbeit
mit GitHub gemacht. Meine CI/CD Projekte  laufen schneller als
je  zuvor  und  dank  der  vereinfachten  Arbeit  kann  ich  meine
Programmierprojekte viel besser skalieren.",
      IMAGE_URL: "images/user1.jpg",
      NAME: "Julia hawkins",
      DESIGNATION: "Programmiererin"
    },
    {
      DESCRIPTION:
        "Früher hatte ich 2 festangestellte, die sich den ganzen
Tag nur um unsere CI/CD Aufgaben gekümmert haben. Der graphische
Editor für GitHub Actions ermöglicht es uns nun, dass die ehemals
für diese Thematik angestellten Mitarbeiter sich auf  kreativere
Aufgaben konzentrieren können.",
      IMAGE_URL: "images/user2.jpg",
      NAME: "John Smith",
      DESIGNATION: "Co-founder Software4You"
    }
  ]
};

const SOCIAL_DATA = {
  HEADING: "Checked  unser Social Media",
  IMAGES_LIST: [
    "images/facebook-icon.png",
    "images/instagram-icon.png",
    "images/whatsapp-icon.png",
    "images/twitter-icon.png",
    "images/linkedin-icon.png",
    "images/snapchat-icon.png"
  ]
};

const FOOTER_DATA = {
  DESCRIPTION:
    "Neugierig geworden? Dann kontaktiere uns! Wir kommen gerne
mit  dir ins Gespräch.",
  CONTACT_DETAILS: {
    HEADING: "Kontaktdaten",
    ADDRESS: "Coblitzallee 6, Mannheim",
    MOBILE: "+49 1562789002",
    EMAIL: "graphischereditor@githubactions.com"
  },
  SUBSCRIBE_NEWSLETTER: "abboniere unseren Newsletter",
  SUBSCRIBE: "Abonnieren"
};

const MOCK_DATA = {
  HEADER,
```

```
  NAVBAR_DATA,
  BANNER_DATA,
  SERVICE_DATA,
  ABOUT_DATA,
  TESTIMONIAL_DATA,
  SOCIAL_DATA,
  FOOTER_DATA
};
export default MOCK_DATA;
```

Die Basis steht nun

Mit dieser Basis kannst du ab jetzt individuell für dich schauen was du von meinem Code nun weiter nutzen möchtest oder nicht. Wie bereits beschrieben habe ich die Module in diesem Projekt in 6 aufgeteilt. Zusätzlich habe ich dir noch ein weiteren Component für eine Navigationsbar mit hinzugefügt. Nutze sie einfach mal zum experimentieren. Mir hat sie ehrlich gesagt so viele Probleme bereitet, dass ich eher privat an ihr weiter arbeiten werde, da sie für diese Landingpage nicht von Relevanz ist.

Komponenten einfügen

Erstelle nun erstmal einen übergeordnet „Components“ Ordner im src. Ordner.

Um meine Komponenten oder Module in dein Projekt einzusetzen kannst du ganz einfach folgende Schritte befolgen. Ich erkläre sie am Beispiel des Banner Komponenten:

Erstelle einen Ordner ,,Banner“ im components Ordner erstelle hier nun einen „File“ „Banner.svelte“. In diesen „File“ kannst du jetzt meinen Code für den Banner aus meine repo kopieren und einfügen.

Wiederhole diese Schritte für alle weiteren Module, die du aus meinem Code nutzen möchtest.

Profis ziehen sich einfach direkt das gesamte Repo ;)

Änderungen machen

Du fragst dich jetzt bestimmt, wie du Änderungen innerhalb des Codes machen kannst, die auch auf deiner Seite erscheinen. Das ist sehr einfach. Du kannst über die Data.js Datei alle Textänderungen vornehmen. Für Jede Textänderung, die du im Banner vornehmen möchtest kannst du einfach die Klasse BANNER DATA bearbeiten. Wenn du beispielsweise den Header ändern möchtest, kannst du ganz einfach in Zeile 12 „HEADING“, den eingetragenen Text ändern.

```
];
const BANNER_DATA = {
  HEADING: "Kein Stress, kein Code und nur Erfolge! Wir revolutionieren GitHub Actions",
  DECRIPTION:
    "Bearbeite jetzt deinen GitHub Actions-Workflow mit  unserem graphischen GitHub Actions
    Editor!",
  TUTORIAL_URL:
    "https://www.youtube.com",
  WATCH_TUTORIAL: "Zum Tutorial"
};
```

Bilder einfügen

Dies kannst du auch genauso einfach mit den Photos machen. Ich empfehle dir, dir erstmal einen Ordner Images im „public“ Ordner zu erstellen. Der Code greift auf die Bilder im ordner zu und alles was du nur noch machen musst ist den Pfad zu den Bildern anzugeben. Hier ein Beispiel:

```
    {
      LABEL: "Wir wir dir das Leben erleichtern",
      DESCRIPTION:
        "Unser graphischer Editor für GitHub Actions gibt dir die volle Kontrolle. Mit dessen
        Hilfe kannst du per Knopfdruck Workflows ausführen, die den Code in deinem Repository
        erstellen und auch Tests darüber ausführen. Konfiguriere deinen CI-Workflow
        beispielsweise so, dass er ausgeführt wird, wenn ein GitHub-Ereignis auftritt, z. B. wenn
        neuer Code in dein Repository übertragen wird. Wir ermöglichen dir sogar einen
        festgelegten Zeitplan einzustellen oder feste Ereignis zu definieren. Das alles ganz ohne
        Code!",
      URL: "images/service9.png"
    }
```

Ziehe dein neues Photo in den Images Ordern und gibt über „URL:“ den direkten Pfad zu deinem Bild an.

Super Easy oder? Ich hoffe ich konnte dir mit meinem Tutorial einen guten „KickOff“ für dein Svelte Projekt bieten. Falls du noch tiefergreifende Fragen hast, bspw. wie die Textstile, Bildpositionierung und was es mit den Klassen auf sich hat, dann schau doch gerne mal in meine „Learnings“ herein. Dort habe ich größere Challenges beschrieben, die ich mit meinen Komilitonen gemeinsam gelöst habe.

Viel Spaß beim coden!