Diplomová práce

SBÍRKA ZNOVUPOUŽITELNÝCH VUE KOMPONENT

Bc. Vojtěch Moravec

Fakulta elektrotechnická
Katedra počítačové grafiky a interakce
Vedoucí: Ing. Oldřich Malec
21. ledna 2024

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů kódu

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisů, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, avšak pouze k nevýdělečným účelům. Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 21. ledna 2024 .....

Abstrakt

Tato práce se zaměřuje na návrh a vývoj sbírky znovupoužitelných komponent pro webové aplikace využívající framework Nuxt. Cílem je vytvořit kolekci komponent, které zjednoduší začátek vývoje a zároveň budou jednoduše rozšiřitelné a upravitelné. Práce se nezaměřuje pouze na samotné komponenty, ale také na jejich dokumentaci a další nástroje podporující celý životní cyklus aplikace od návrhu až po nasazení.

K dispozici jsou i rozsáhlejší sekce, skládající se z těchto komponent. Vývojáři tak mohou využít již předpřipravená řešení, která si mohou dle potřeby upravit, což značně urychluje proces vývoje.

Pro designéry je připravena knihovna ve Figmě, umožňující jim poskytovat vývojářům návrhy, které lze replikovat v přesné shodě s originálem. Vývojáři mohou využít několik způsobů instalace komponent a to sice pomocí kopírování kódu nebo pomocí CLI.

Klíčová slova UI, UX, DX, Nuxt, Vue, komponenty, sbírka komponent, znovupoužitelnost

Abstract

This thesis focuses on the design and development of a collection of reusable components for web applications using the Nuxt framework. The goal is to create a collection of components that simplify the start of development and are easily expandable and modifiable. The work focuses not only on the components themselves but also on their documentation and other tools supporting the entire application lifecycle from design to deployment.

More extensive sections composed of these components are also available. Developers can thus utilize pre-prepared solutions, which they can modify as needed, significantly accelerating the development process.

For designers, a library in Figma has been prepared, allowing them to provide developers with designs that can be replicated in exact accordance with the original. Developers have several ways to install components, either by copying code or using a CLI.

Keywords UI, UX, DX, Nuxt, Vue, components, collection of components, reusable

Seznam zkratek

CLI	Command Line Interface
CSS	Cascading Style Sheets
FE	Frontend
IT	Informační technologie
JS	JavaScript
JSON	JavaScript Object Notation
NPM	Node Package Manager
PC	Personal Computer
SPA	Single Page Application
SSR	Server Side Rendering
UI	User interface
URL	Uniform Resource Locator

Slovník

Build	Sestavení aplikace do finální podoby
Deploy	Nasazení aplikace do testovacího či prostředí
Frontend	Část aplikace, kterou vidí uživatel a reaguje s ní
Hosting	Služba hostování webové stránky
Hover	Najetí myši na prvek
Pop-up	Vyskakovací okno
Pull request	Žádost o změnu ve zdrojovém kódu
Push	Nahrání změněného kódu do repozitáře
Wireframe	Drátěný model

Kapitola 1
Úvod

UI knihovny jsou nezbytné nástroje ve světě webového vývoje. Poskytují sadu předdefinovaných komponent, které vývojářům umožňují rychle a efektivně vytvářet atraktivní a funkční uživatelská rozhraní. Tyto knihovny jsou důležitou součástí vývoje aplikací, protože usnadňují implementaci složitých UI prvků.

Tato práce se zabývá analýzou různých aspektů UI knihoven, včetně jejich designu, funkcí, a toho, jak tyto knihovny podporují různé části vývoje aplikace. Cílem je poskytnout ucelený pohled na UI knihovny, jejich výhody, výzvy a *best practices* pro jejich využití ve webovém vývoji.

Kapitola 2

Cíl

Cílem této práce je vytvořit sadu znovupoužitelných a snadno modifikovatelných komponent pro webové aplikace postavené na frameworku Nuxt. Tato kolekce komponent má usnadnit počáteční fázi vývoje, nabízet možnosti rozšíření a přizpůsobení podle individuálních potřeb. Práce se neomezuje pouze na vývoj samotných komponent, ale zahrnuje také jejich podrobnou dokumentaci a nástroje, které podporují všechny fáze tvoření aplikace.

Nejdříve je v práci připravena teoretická část, která se zaměřuje na popis zvolených technologií a dalších řešení. Dále se práce zabývá návrhem a vývojem komponent a jejich dokumentace. Nakonec je nastíněn možný budoucí vývoj a rozšíření.

Kapitola 3
Analýza

3.1 Kvalitativní průzkum

Vzhledem k tomu, že se jedná o projekt, který je určený hlavně pro využití autora, není kvalitatní průzkum příliš relevantní. Místo toho se práce zaměřuje na porovnání existujících řešení, jejich výhod a nevýhod a názorů komunity. Nicméně v jedné z následujících kapitol jsou výsledky z testování a výčet následných úprav.

3.2 Existující řešení

Existující řešení lze rozdělit do dvou skupin. První skupinou jsou knihovny, které nabízejí sadu komponent jako balíček, který se do projektu přidá jako závislost. Poté se komponenty importují a používají v kódu. Jejich stylování lze upravit, ale většinou pouze pomocí předpřipravených proměnných nebo limitovaným rozhraním. Úprava jejich funkcionality je většinou velmi omezená. Jakmile chce uživatel upravit větší části, musí přistoupit k ohybání kódu, což často vede k neudržitelnému kódu.

Druhou skupinou jsou knihovny, které nejsou připraveny pro použití jako balíček. Jejich kód je dostupný pro zkopírování do projektu, popř. mají dostupné CLI, které umožňuje komponenty přidat pomocí jednoduchého příkazu. Tato skupina míří na lepší znovupoužitelnost kódu a zároveň na vyšší možnosti rozšíření a úprav.

3.2.1 Knihovny s principem balíčkování

3.2.1.1 NuxtUI

Jako zástupce knihoven na principu balíčkování jsem si vybral NuxtUI. Tuto knihovnu jsem využil na několika projektech, takže jsem si dokázal udělat představu o jejích výhodách a nevýhodách. Líbilo se mi, že nabízí velké množství komponent, které jsou propracované a hezky nastylované. Na druhou stranu jsem se často potýkal s tím, že jsem chtěl upravit funkčnost komponenty, ale nebylo to možné. Většinou jsem musel přistoupit k vytvoření ohnutí ruzných částí. Tyto úpravy byly často velmi složité a neudržitelné.

- **Výhody**
  - Velké množství komponent
  - Propracované styly
  - Připravené pro použití
- **Nevýhody**
  - Omezené možnosti úprav
  - Složité úpravy stylů
  - Složité úpravy funkcionality

3.2.2 Knihovny s principem vlastnění kódu

Knihovny, kde výsledný kód vlastníte většinou obsahují komponenty, které jsou jednodušší. Jejich výhodou je, že je lze snadno upravit, protože kód není schovaný v balíčku. Nevýhodou je, že některé složitejší operace si uživatel musí vytvořit sám. Je tu ale jedna vyjímka, na kterou se chci zaměřit, protože na podobném principu chci stavět i svojí kolekci.

3.2.2.1 Radix UI

RadixUI představuje open-source knihovnu UI komponent, která je zaměřena na tvorbu kvalitních a přístupných designových systémů a webových aplikací. Jde především o Radix Primitives, což jsou nízkoúrovňové UI komponenty s důrazem na přístupnost a možnost úprav vývojářů za cílem vytvoření vlastní knihovny. Tyto komponenty lze využívat buď jako základní vrstvu designového systému nebo je postupně implementovat do stávajících projektů. [1]

3.2.2.2 shadcn/ui

Shadcn/ui je inovativní open-source knihovna UI komponent, která je navržena tak, aby vylepšila webový vývoj, zejména pro projekty využívající React. Hlavní předností této knihovny je její lehkost, díky čemuž se snadno integruje do projektů bez nutnosti zatěžujících závislostí. Knihovna staví zejména na komponentách Radix UI, které kladou důrat na přístupnost, což zajišťuje, že komponenty jsou inkluzivní a použitelné pro všechny uživatele. [2]

Dalším významným aspektem knihovny Shadcn/ui je integrace s frameworkem Tailwind CSS, který poskytuje efektivní a přívětivé prostředí pro vývojáře, preferující tento CSS framework orientovaný na utilitu. Výhody Tailwind CSS jsou popsány v kapitole věnované technologiím. Tato integrace umožňuje snadnou úpravu a rozšíření stylů. Shadcn/ui vyniká svou snadnou použitelností a detailní kontrolou nad komponentami. Vývojáři mohou přímo přistupovat ke zdrojovému kódu jednotlivých komponent, což umožňuje jejich efektivní úpravy pro specifické případy užití a požadavky aplikace.

Ruční instalace nebo kopírování každé komponenty může být pro některé vývojáře zdlouhavé, zejména pro ty, kteří jsou zvyklí importovat komponenty z balíčků. Ačkoli přímý přístup ke kódu komponent prospívá modularitě a rozšiřitelnosti, může vést k rozšíření objemu kódu.

- **Výhody**
  - Jednoduché úpravy
  - Kód vlastní uživatel
  - Aktualizace knihovny neovlivní kód
- **Nevýhody**
  - Omezená funkcionalita
  - Větší objem spravovaného kódu

Kapitola 4
Technologie

4.1 Nuxt

Nuxt je moderní a výkonný open-source framework založený na Vue, který je navržen pro intuitivní a efektivní vývoj webových aplikací a stránek. Tento framework se vyznačuje tím, že automaticky řeší mnoho opakujících se úkolů vývojářů, což jim umožňuje soustředit se na tvorbu samotné aplikace. Nuxt využívá konvence a strukturované adresářové uspořádání pro automatizaci procesu vývoje a nabízí možnost vlastní konfigurace a přizpůsobení výchozích chování.

Nuxt má narozdíl od čistého Vue schopnosti server-side renderingu (SSR). Tato funkce zajišťuje rychlejší načítání stránek a lepší SEO optimalizaci, protože celý obsah stránky je serverem vygenerován dříve, než začne běžet klientský JavaScript. Nuxt tento přístup kombinuje v takzvaném hybrid renderingu a má tedy výhody tradičního server-side renderingu s interaktivitou a pokročilými uživatelskými rozhraními single-page aplikací (SPA). [3]

4.2 Tailwind

Tailwind je moderní a oblíbený CSS framework, který se zaměřuje na tzv. utility-first přístup. Tento přístup umožňuje vývojářům rychle a efektivně vytvářet vlastní komponenty a designy s použitím nízkoúrovňových CSS tříd, které jsou přímo integrovány do HTML markupu.

Jednou z hlavních výhod Tailwind je možnost psát méně vlastního CSS. Vývojáři mohou využívat předdefinované třídy, jako jsou flexbox a padding utility, což znamená, že většina stylů je opakovaně použitelná a zřídka je potřeba psát nové CSS. Tailwind také eliminuje potřebu vymýšlet názvy tříd, protože vývojáři vybírají třídy z předdefinovaného designového systému. To znamená, že nemusí přemýšlet o ”dokonalých”názvech tříd pro určité styly a komponenty nebo si pamatovat složité názvy. [4]

Tailwind je známý svou flexibilitou a kontrolou nad tím, jak aplikace vypadá, což poskytuje větší prostor pro vytváření jedinečných webů. Na rozdíl od jiných CSS frameworků, jako je Bootstrap nebo Materialize, Tailwind nenabízí plně stylované komponenty, jako jsou tlačítka nebo dropdown menu. Místo toho nabízí užitečnostní třídy, které umožňují vytvořit vlastní opakovaně použitelné komponenty.

Přestože Tailwind nabízí mnoho výhod, může být jeho použití obtížné pro ty, kteří nejsou zkušení s CSS, a může způsobovat zmatek kvůli množství informací uložených v HTML souboru. Navíc při instalaci Tailwindu jsou výchozí CSS styly odstraněny, takže je nutné je pro všechny elementy znovu vytvořit.

Kapitola 5
Implementace

Začal jsem vytvořením projektu pomocí `npx nuxi@latest init content-app -t content`. Tento příkaz vytvoří projekt s připravený Nuxt Content modulem. Dále jsem přidal potřebné moduly pro vývoj, jako je například Tailwind, Nuxt Icon, Nuxt Color Mode a v neposlední řadě Nuxt SEO. Pro větší konzistenci kódu jsem přidal i ESLint a Prettier.

5.1 Nastavení Tailwindu

Nejdříve jsem se potýkal s chybějícími styly. Chtěl jsem to řešit pomocí *safelistu*, ale to mi nepřišlo jako ideální řešení. Po chvíli jsem zjistil, že lze specifikovat, které složky má při buildu Tailwind procházet a hledat tak potřebné styly. Tailwind totiž využívá tree shakingu, takže výsledný balíček obsahuje pouze styly, které jsou použité. Toto nastavení jsem přidal do `tailwind.config.js`. Konkrétně se jednalo o řádek 7 v ukázce 1.

```
1  import type { Config } from 'tailwindcss'
2  import typography from '@tailwindcss/typography'
3  import containerQueries from '@tailwindcss/container-queries'
4
5  export default <Partial<Config>>{
6      darkMode: 'class',
7      content: ['components/**/*.{vue,ts}', 'layouts/**/*.vue',
       ↪  'pages/**/*.vue'],
8      theme: {
9          extend: {
10             colors: {
11                 background: 'hsl(var(--background))',
12                 primary: {
13                     DEFAULT: 'hsl(var(--primary))',
14                     contrast: 'hsl(var(--primary-contrast))',
15                 },
16                 secondary: {
17                     DEFAULT: 'hsl(var(--secondary))',
18                     contrast: 'hsl(var(--secondary-contrast))',
19                 },
20                 disabled: {
21                     DEFAULT: 'hsl(var(--disabled))',
22                     contrast: 'hsl(var(--disabled-contrast))',
23                 },
24                 border: 'hsl(var(--border))',
25             },
26             borderRadius: {
27                 sm: 'calc(var(--radius) - 4px)',
28                 md: 'calc(var(--radius) - 2px)',
29                 lg: 'var(--radius)',
30                 xl: 'calc(var(--radius) + 4px)',
31                 '2xl': 'calc(var(--radius) + 8px)',
32                 '3xl': 'calc(var(--radius) + 16px)',
33             },
34         },
35     },
36     plugins: [typography, containerQueries],
37 }
```

Výpis kódu 1 Konfigurační soubor pro Tailwind

5.2 Zobrazení kódu

Nuxt Content má zabudovanou možnost zobrazení kódu. [5] Potřeboval jsem vymyslet, jak zobrazit kód bez jeho duplikace. V tom mi pomohl zdrojový kód knihovny NuxtUI. Tato knihovna využívá principu vygenerování endpointů pro zobrazení zdrojového kódu přímo ze souborů s komponentami, popřípadě ze souborů s ukázkami použití. [6]

5.3 Deploy

Dokumentace se automaticky nasadí pomocí Github Actions na Cloudflare Pages. Nastavení je velmi jednoduché, stačí mít účet na Githubu a Cloudflare, po vytvoření repozitáře jej spojit s projektem na Clouflare Pages a při každém pushi do větve `main` se dokumentace automaticky nasadí.

Kapitola 6

Testování

Testování proběhne v rámci diplomové práce. Bude se zaměřovat na několik oblastí. První bude testování samotných komponent. Dále bude testována dokumentace a její přehlednost. Poslední oblastí bude testování instalace a použití komponent.

Kapitola 7

Možnosti rozšíření

V rámci diplomové práce budu pokračovat v rozšiřování kolekce komponent. Také rozšířím možnost instalace o CLI, které umožní jednodušší přidávání komponent do projektu. Určitě bude důležité přidat více možností úprav stylů a témat. Kromě toho by se mi líbilo mít několik stylů dostupných na výběr.

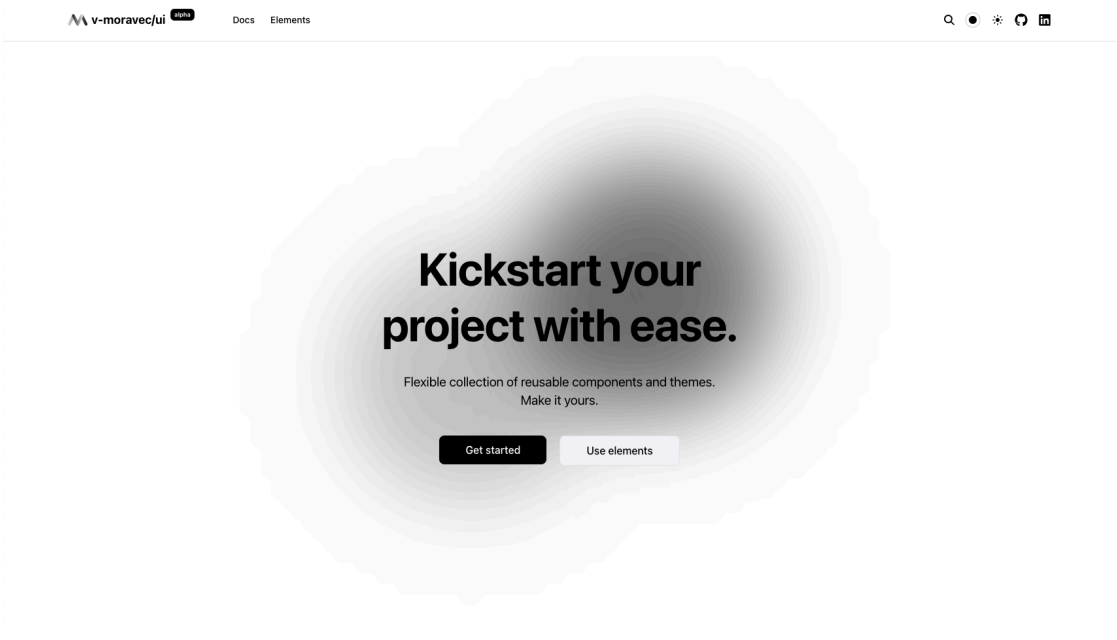
Kapitola 8
Závěr

Cílem práce bylo analyzovat současný stav komponentových knihoven a na základě toho navrhnout a implementovat vlastní řešení. Povedlo se mi vytvořit ekosystém, kam budu moci v rámci diplomové práce přidávat další komponenty. Mezi části tohoto ekosystému patří dokumentace, která je dostupná na adrese `https://ui.vojtamoravec.cz`. Dále se jedná o knihovnu komponent ve Figmě. Poslední část je ještě neimplementované CLI, které bude pomáhat s instalací komponent. Kromě samotných komponent počítám ještě s vylepšením celého ekosystému.

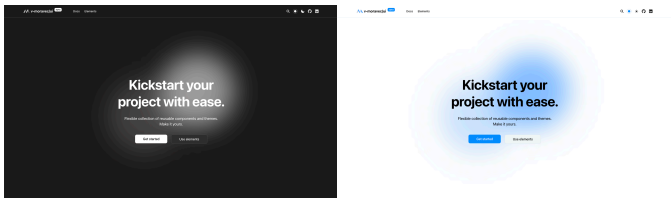
Příloha A

Snímky dokumentace

A.1 Úvodní obrazovka



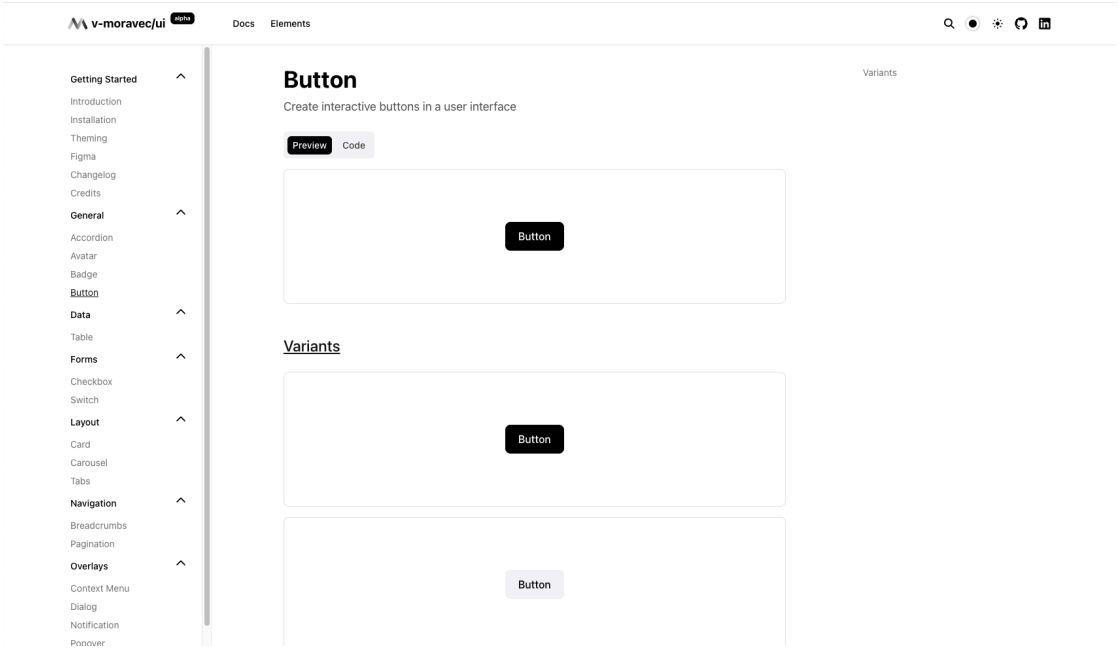
Obrázek A.1 Úvodní obrazovka



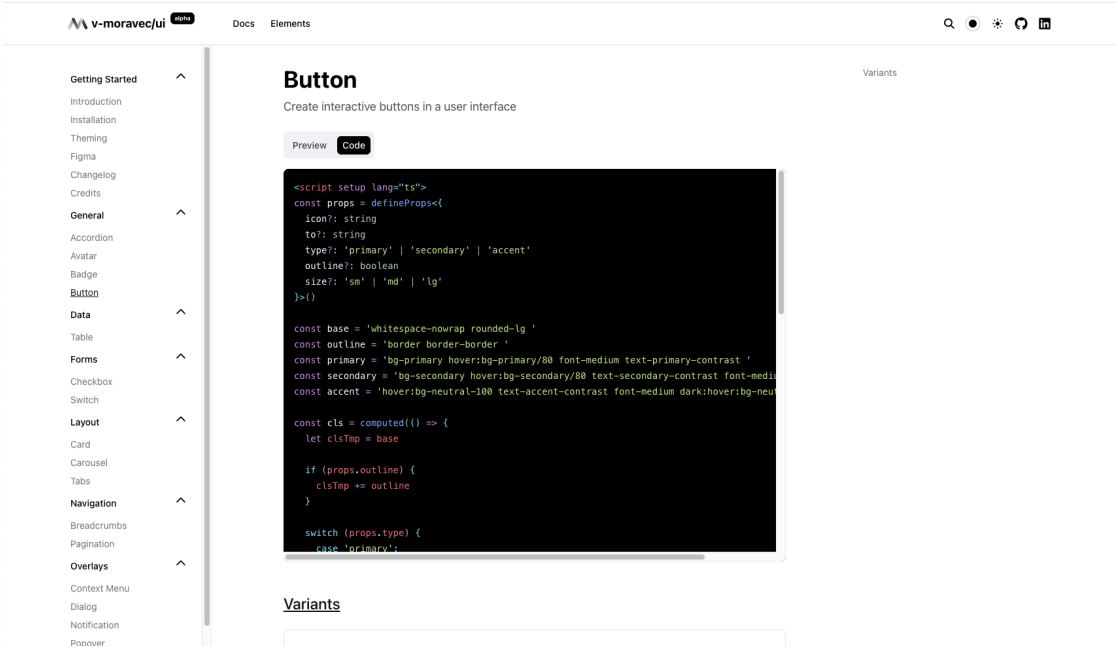
(a) Tmavá verze **(b)** Modrá verze

Obrázek A.2 Stylování

A.2 Komponenty

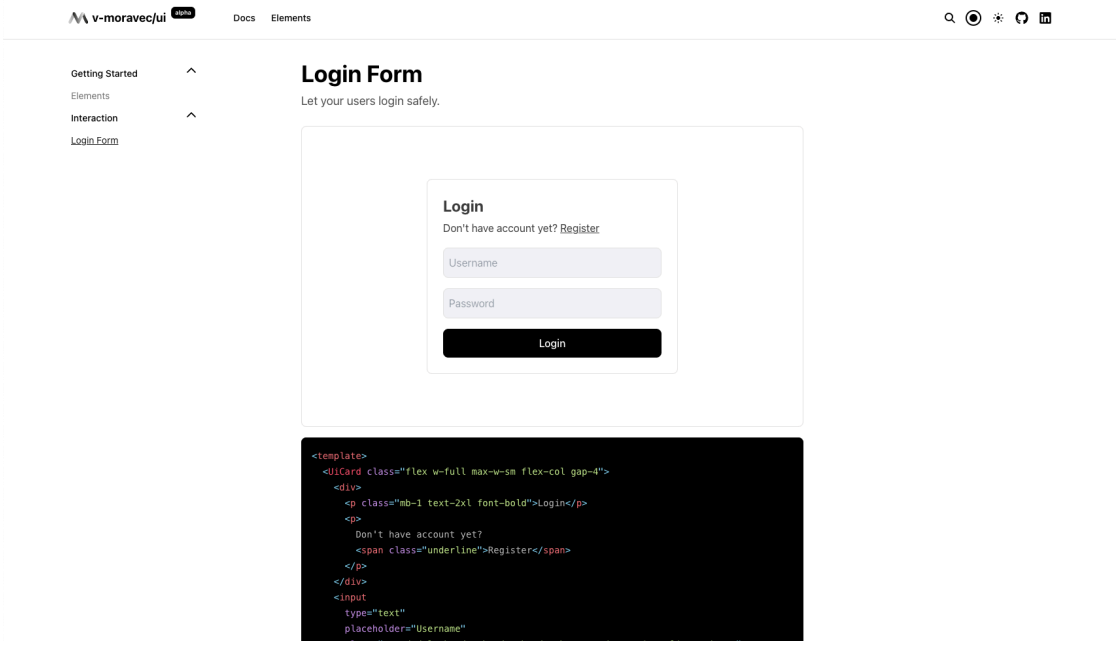


Obrázek A.3 Zobrazení komponenty



Obrázek A.4 Kód komponenty

A.3 Elementy



Obrázek A.5 Element složený z komponent

Zdroje

1. *Radix UI* [online]. Radix UI [cit. 2024-01-21]. Dostupné z: `https://www.radix-ui.com/primitives/docs/overview/introduction`.
2. *shadcn/ui* [online]. shadcn [cit. 2024-01-21]. Dostupné z: `https://ui.shadcn.com/docs`.
3. *Rendering Modes* [online]. Nuxt [cit. 2024-01-21]. Dostupné z: `https://nuxt.com/docs/guide/concepts/rendering`.
4. *Utility-First Fundamentals* [online]. Tailwind [cit. 2024-01-21]. Dostupné z: `https://tailwindcss.com/docs/utility-first`.
5. *Code Highlighting* [online]. Nuxt [cit. 2024-01-21]. Dostupné z: `https://content.nuxt.com/usage/markdown#code-highlighting`.
6. *Code Highlighting* [online]. Nuxt [cit. 2024-01-21]. Dostupné z: `https://github.com/nuxt/ui/blob/dev/docs/modules/content-examples-code.ts`.