WAP - Internetové aplikace
2022 / 2023

Projekt 2

Webový frontend využívající dostupná otevřená data

Dopravní nehody s chodci v okolí škol a sportovišť v Brně

Autoři: Jan Lorenc (xloren15)
David Holas (xholas11)
Martin Konečný (xkonec79)

Datum: 14. 4. 2023

1. Zadání

Úkolem projektu bylo vymyslet a vytvořit webový frontend k libovolným otevřeným datům. Za data jsme si zvolili základní školy a sportoviště v Brně z portálu data.brno.cz a chtěli jsme zobrazit a analyzovat dopravní nehody s účastí chodců v jejich blízkosti. Požadavky na aplikaci schválené vedoucím projektu bylo zobrazení škol a sportovišť formou tabulky s možností filtrace, mít možnost je ukázat na mapě s nehodami v jejich blízkosti a dále formou grafů vyobrazit základní statistiky

2. Návrh

Vzhledem k tomu, že data nemusí být vždy dostupná nebo se odkaz na ně může změnit, je vhodné vytvořit vlastní webové API, které je by je zprostředkovávalo způsobem, který máme pod kontrolou. Periodicky si data aktualizuje z data.brno.cz a v případě nedostupnosti či změny odkazu poskytne alespoň svou poslední verzi dat.

Frontendová aplikace pak kromě úvodní a chybové stránky obsahuje stránky zvlášť pro školy a zvlášť pro sportoviště (dále v obecnosti jen instituce). Ve své podstatě se moc neliší, a proto je vhodné mít jednotlivé části rozdělené na komponenty a tyto přepoužít jen pro jiná data. Pro takový návrh se hodí knihovna či framework zaměřený právě na komponenty, jako je například React, který je i použit. Tabulka s institucemi umožňuje export dat, řazení dle sloupců, fulltextové vyhledávání a jeden vybraný filtr. Při zvolení instituce se tato zobrazí na mapě spolu s nehodami v její blízkosti. Dále lze filtrovat nehody dle vzdálenosti od instituce a dle datumu. Je možné zobrazit i všechny instituce s jejich nehodami, ne pouze zvolenou, nicméně to může vést k nepřehledné mapě a zvýšení latence aplikace vzhledem k velkému množství geografických dat. Dle zvolených filtrů jak institucí, tak nehod jsou v grafech zobrazeny statistiky o počtu zranění či úmrtí a zda se jedná o děti či dospělé.

3. Implementace

3.1. Web API

K tvorbě API byl použit framework .NET 6 a Swagger k jeho snadné vizualizaci. Jedná se o REST API a poskytuje celkem 6 endpointů, 2 pro každou z domén nehody, školy a sportoviště. Každá doména má endpoint na stažení datasetu a aktualizaci z data.brno.cz. Odkazy ke stažení dat a cesta k lokálnímu uložení je konfigurovatelná v souboru appsettings.json. API datasety nijak nemění, pouze uchovává. Při spuštění aplikace se automaticky data stáhnou/aktualizují a je naplánována aktualizace vždy na půlnoc každý den pomocí knihovny Quartz.

3.2. Frontend aplikace

Aplikace je napsána v Reactu a Typescriptu s použitím UI knihovny MUI, tedy Material UI. Nejedná se o SPA aplikaci, nýbrž je rozdělena na samostatné stránky popsány dále. Kód aplikace má následující strukturu:

- */public* – Obsahuje základní index.html a statické obrázky pro aplikaci.
- */src* – Zahrnuje React zdrojové kódy stránky.
  - *index.tsx* – Vyrenderuje React aplikaci do html stránky.
  - *app.tsx* – Poskytuje kořenovou komponentu aplikace. Ta navíc řeší routing a barevné téma.
  - */styles* – Obsahuje css styly pro jednotlivé stránky či komponenty. Dále jsou zde nadefinovány barvy pro téma aplikace.
  - */data/downloader.ts* – Poskytuje funkci ke stažení a konverzi dat z REST API.
  - */pages* – Obsahuje komponenty pro jednotlivé stránky. Za zmínku stojí pouze stránky pro instituce. Ty se starají o stažení dat institucí, jejich filtraci a zobrazení. Dále vykreslují velmi důležitou komponentu řešící nehody. Více o samostatných komponentách je řečeno dále.

- */controls* – V této složce se nachází kontrolky/komponenty použité na stránkách.
  - *search-field.tsx* – Rozšíření MUI TextFieldu fungující jako fulltextový SearchField.
  - *date-range-picker.tsx* – Range picker je pouze v placené verzi MUI, proto jsme si vytvořili vlastní pomocí DatePickerů.
  - *table.tsx* – Wrapper nad MUI DataGridem pro redukci boilerplate kódu. Navíc řeší i výběr instituce.
  - *accidents.tsx* – Jedná se o komponentu zobrazující informace o nehodách. V sobě zahrnuje mapu s filtrem nehod a grafy.
  - *charts*.tsx – Komponenta, která má zodpovědnost za výpočet a zobrazení grafů nehod.
  - */maps* – Obsahuje komponenty pro mapu, jako jsou tooltipy, markery a samotnou mapu, pro kterou jsme použili google mapy.
- */assets* – Statické soubory, jmenovitě svg markery do mapy

4. Spuštění

Webová aplikace spoléhá na běžící API pro stažení dat. To lze spustit velice snadno příkazem `dotnet run` z adresáře */src/api/WapApi*. Všechny závislosti se stáhnou a zkompilují automaticky. Frontend aplikace je spravována create-react-app a lze ji spustit příkazem `npm [run] start`. Předtím je však nutné nainstalovat závislosti z *package.json* pomocí `npm install`. Oba tyto příkazy lze spustit z adřesáře */src/app*, kde se nachází *package.json* konfigurační soubor.

5. Použité technologie a knihovny

- **.NET 6** – Pro webové REST API. S tímto frameworkem máme největší zkušenosti a jedná se o kvalitní technologii pro tvorbu API.
- **React** – Pro webovou aplikace. Jedná se o jednu z nejpopulárnějších frontendových knihoven a hodí se ke komponentnímu vývoji.
- **Typescript** – Nebyla nutnost, ale chtěli jsme si vyzkoušet zařazení typů do JS vývoje.
- **MUI** – Rozšířená UI knihovna poskytující potřebné kontrolky jako tabulka, textové pole apod. Zároveň podporuje barevné téma, stylování a layout.
- **google-maps** – Pro zakomponování snippetu mapy jsme použili Google mapy, jelikož mají rozsáhlejší a lépe dokumentované api, než jiné nalezené knihovny.
- **recharts** – K vykreslování grafů. Zvoleno pro rozšířenost, podporu a potřebnou funkcionalitu.
- **axios** – Populární knihovna pro zasílání požadavků (využito k volání api)
- **csvtojson** – Tato knihovna byla využita ke konverzi csv datasetu do JS objektů
- **dajjs** – MUI datepickery vyžadují nějakou časovou knihovnu pro svou funkcionalitu. Ve srovnání například s momentem je dajjs odlehčenější, a proto byla použita právě tato.
- **emotion** – Povinná závislost pro MUI, neboť je emotion použit pro MUI stylování.
- **react-router-dom** – Jelikož jsme se rozhodli rozdělit aplikaci na stránky a nenechat ji jako SPA, potřebovali jsme vyřešit směrování stránek, o což se stará tato knihovna.

6. Rozdělení práce

- Jan Lorenc – REST API, layout aplikace, tabulka a její filtry
- David Holas – Integrace mapy, zobrazení nehod a filtrace
- Martin Konečný – Statistiky, grafy, layout

7. Zhodnocení

Na projektu jsme pracovali průběžně a s časovým předstihem. Díky tomu a taky dobré komunikaci se nám dařilo v klidu řešit všechny úkoly a problémy. Byli jsme poněkud zklamaní z MUI, kde jsme se potýkali s tím, že některé potřebné funkcionality (date range picker, některé vlastnosti datagridu) jsou dostupné pouze v Pro verzi. Největší výzvou byly google mapy, zejména v oblasti přehlednosti a výkonu, nicméně i s tím jsme si dokázali poradit. Obzvlášť pěknou funkcionalitou je shlukování nehod a institucí na mapě. Z vytyčených cílů jsme zvládli implementovat vše a projekt hodnotíme jako úspěšný. Naučili jsme se pracovat s novými technologiemi a knihovnami a rozhodně vidíme v projektu přínos do budoucna.