FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA INFORMATIKY
A VÝPOČETNÍ TECHNIKY

Bakalářská práce

Vývoj Javascript knihovny pro embedování vizualizací skrze Emplifi Public API

Milan Janoch

PLZEŇ

2023

FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA INFORMATIKY
A VÝPOČETNÍ TECHNIKY

Bakalářská práce

Vývoj Javascript knihovny pro embedování vizualizací skrze Emplifi Public API

Milan Janoch

Vedoucí práce
Doc. Ing. Dalibor Fiala, Ph.D.

PLZEŇ 2023

Citace v seznamu literatury:
JANOCH, Milan. *Vývoj Javascript knihovny pro embedování vizualizací skrze Emplifi Public API*. Plzeň, 2023. Bakalářská práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra informatiky a výpočetní techniky. Vedoucí práce Doc. Ing. Dalibor Fiala, Ph.D.

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2023/2024

Studijní program: Informatika a výpočetní technika
Forma studia: Prezenční
Specializace/kombinace: Informatika (INF18bp)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

Jméno a příjmení:	**Milan JANOCH**
Osobní číslo:	**A21B0152P**
Adresa:	V Brance 20, Přeštice, 33401 Přeštice, Česká republika
Téma práce:	Vývoj Javascript knihovny pro embedování vizualizací skrze Emplifi Public API
Téma práce anglicky:	
Jazyk práce:	Čeština
Vedoucí práce:	Doc. Ing. Dalibor Fiala, Ph.D. Katedra informatiky a výpočetní techniky

Zásady pro vypracování:

1. Prostudujte problematiku integrování analytických grafů do aplikací třetích stran v kombinaci se zabezpečeným přístupem přes OAuth 2.
2. Navrhněte knihovnu pro komunikaci s public API od firmy Emplifi včetně zabezpečeného přístupu.
3. Implementujte navrženou knihovnu.
4. Řešení řádně otestujte a minimálně kritické části testujte i s využitím unit testů.
5. Vytvořené řešení kriticky zhodnoťte.

Seznam doporučené literatury:

dodá vedoucí bakalářské práce

Podpis studenta: Datum:

Podpis vedoucího práce: Datum:

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného akademického titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Západočeská univerzita v Plzni má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Plzni dne 15. prosince 2023

. . . . . .
Milan Janoch

V textu jsou použity názvy produktů, technologií, služeb, aplikací, společností apod., které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

Abstrakt

Tato bakalářská práce se zaměřuje na vývoj specializované JavaScript knihovny s cílem umožnit snadné embedování vizualizací do aplikací třetích stran. Hlavními cíli práce jsou návrh a implementace knihovny, navržení rozhraní pro efektivní komunikaci s Public API firmy Emplifi a zajištění bezpečného přístupu k datům pomocí OAuth 2 protokolu. V teoretické části práce je diskutována problematika spojená s embedováním vizualizací do externích aplikací, bezpečný přístup k datům třetích stran a jsou analyzována již existující řešení. Praktická část se zaměřuje na návrh a implementaci JavaScript knihovny, popisuje navržení rozhraní pro efektivní komunikaci s API a zabývá se implementací bezpečného přístupu k datům v souladu se standardem OAuth 2.

Abstract

This bachelor thesis focuses on the development of a specialized JavaScript library to enable easy embedding of visualizations into third-party applications. The main goals of the thesis are to design and implement the library, design an interface to communicate efficiently with Emplifi's Public API and provide secure data access using the OAuth 2 protocol. The theoretical part of the thesis discusses the issues related to embedding visualizations in external applications, secure access to third-party data and analyzes existing solutions. The practical part focuses on the design and implementation of a JavaScript library, describes the design of interfaces for efficient communication with APIs and deals with the implementation of secure data access in accordance with the OAuth 2 standard.

Klíčová slova

OAuth 2.0 • embedování • Emplifi Public API • token • JavaScript

Poděkování

Na tomto místě bych rád poděkoval všem, kteří přispěli k úspěšnému dokončení této bakalářské práce. Velké díky patří především

Bc. Ondřejovi Altmanovi za trpělivost, cenné rady a vstřícnost při navrhování a implementaci praktické části

Ing. Michalovi Kacerovskému za poskytování připomínek v rámci detailní revize kódu, která významně přispěla k vylepšení čitelnosti a efektivity kódu

Monice Opltové za důkladné a pečlivé otestování implementované praktické části

Doc. Ing. Daliborovi Fialovi, Ph.D. za jeho spolupráci a ochotu při tvorbě teoretické části

a také rodině za podporu během celého studia.

Obsah

1 Úvod **3**

2 Embedded analytics a zabezpečení **5**

2.1 Princip 5

2.1.1 Iframe 6

2.1.2 Webová komponenta 7

2.1.3 React SDK + API 7

2.2 Zabezpečení 7

2.2.1 HTTP autentizace 8

2.2.2 API key 8

2.2.3 OAuth 8

2.3 Existující řešení 10

2.3.1 GoodData 10

2.3.2 Microsoft Power BI 10

2.3.3 Powered by Looker 11

2.3.4 Tableau 11

3 Návrh knihovny a uživatelského rozhraní **13**

3.1 Technologie 13

3.1.1 JavaScript a React 13

3.1.2 Emplifi API 14

3.1.3 PreJSON a Vision 18

3.2 Návrh knihovny 19

3.2.1 Omni Studio 19

3.2.2 Funkcionalita knihovny 20

3.3 Návrh UI 21

3.3.1 Vytváření tokenů 21

3.3.2 Náhled vizualizací 22

3.4 Dokumentace 22

4 Implementace **25**
4.1 Implementace knihovny 25
4.1.1 25
4.2 Implementace uživatelského rozhraní 25

5 Závěr **27**

Bibliografie **29**

Seznam obrázků **33**

Seznam tabulek **35**

Seznam výpisů **37**

Úvod 1

V současné době internetu tvoří data významnou a důležitou součást každodenního života. Problematikou dat je nejen je efektivní ukládání, ale také rychlé a efektivní interpretování. Vizualizace dat není pouhým trendem, ale stala se klíčovým nástrojem v mnoha odvětvích. Od oblasti logistiky, kde pomáha monitorovat a řídit tok zásob a logistické operace, až po oblast sociálního marketingu, kde je využívána na k personalizovaným reklamám či sledování aktivit zákazníků. Pro snadné integrování vizualizací do aplikací třetích stran se využívá koncept embedded analytics – ten umožňuje uživatelům rychlý a efektivní přístup k datům bez potřeby přecházet mezi různými aplikacemi.

Cílem této práce, zadanou společností Emplifi, která se specializuje na sociální marketing, je vytvořit praktickou knihovnu umožňující embedování analytických grafů do aplikací třetích stran. Velký důraz bude kladen na bezpečný přístup k datům skrze OAuth 2 protokol, který je klíčovým prvkem Emplifi Public API. Součástí bude také administrační rozhraní, jenž umožní snadné generování a obnovování OAuth 2 tokenu, a podrobná uživatelská dokumentace, která bude obsahovat veškeré informace k nastavení a používání knihovny či uživatelského rozhraní.

V rámci teoretické části bude podrobně diskutována problematika spojená s embedováním. Analyzovat se bude princip, výhody a nevýhody bezpečnostního protokolu OAuth 2 a již existující řešení embedded analytics.

Tímto způsobem práce nespojuje pouze technologickou inovaci s nutností bezpečnosti dat, ale také reaguje na aktuální potřeby v odvětví sociálního marketingu a digitálního prostoru.

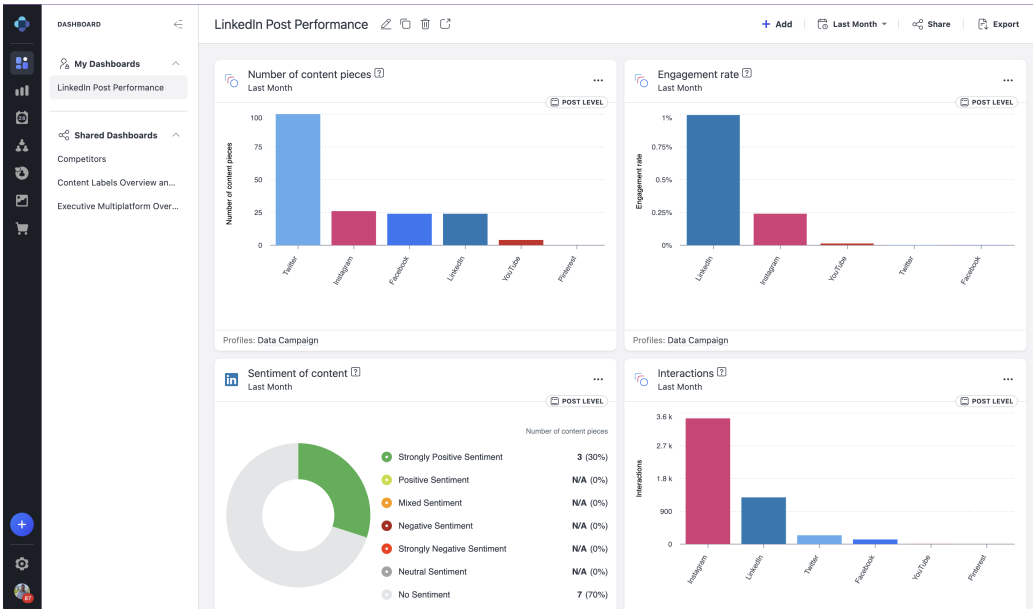
Embedded analytics a zabezpečení — 2

V této kapitole se zmíníme o principu embedded analytics, zabezpečení dat pomocí OAuth 2 tokenu a již existujících řešeních.

2.1 Princip

Embedded analytics transformuje data do grafů a dashboardů [1]. Dashboard je souhrn informací a nástrojů, který umožňuje rychle a jednoduše zobrazovat důležité metriky (např. počet komentářů pod příspěvkem) [2]. Emplifi dashboardy umožňují definovat, vytvářet a vizualizovat aktivity uživatelů na sociálních sítí [3], což umožňuje komplexní pohled na interakci s obsahem.

Na obrázku 2.1 vidíme příklad dashboardu - skládá se z několika grafů, obvykle



Obrázek 2.1: Ukázka dashboardu v produktu firmy Emplifi [4]

nazývané jako widgety [3], jejichž obsah lze modifikovat pomocí přepínačů v liště. Obsah může být modifikován v časovém období (posledních 30 dní, poslední rok, konkrétní časový rozsah, ...), ale lze jej modifikovat také na základě metrik jako je např. sociální síť (Facebook, Instragram, LinkedIn), druh interakcí (komentáře, liky, sdílení) apod.[1]

Embedded analytics se nechová a nevypadá jako samostatná aplikace, ale je integrován do jiného softwaru nebo webového portálu. Koncoví uživatelé často ani nepoznají, že se jedná integrovanou analytiku a vnímají software s touto integrovanou analytikou jako jeden nástroj [1]. To umožňuje softwarovým společnostem získat a plně integrovat analytickou platformu s jejich vlastním SaaS produktem bez nutnosti velkých investic do vývoje vlastního řešení.

SaaS (Software as a Service) je licenční model, v němž přístup je poskytován na základě předplatného, přičemž software je umístěn na externích serverech, nikoli na firemních serverech [5]. K těmto službám se běžně přistupuje prostřednictvím webového prohlížeče s přihlašovacím jménem a heslem. Výhodou je, že místo toho, aby mělo každé zařízení ve firmě nainstalované tento program, stačí se do programu přihlásit přes internet. Pro firmu to znamená úsporu financí, jelikož nemusí investovat do nového hardwaru, na němž by dané programy běžely. Nevýhodou SaaS je bezpečnost dat a rychlost jejich doručování. Protože jsou data umístěna na externích serverech, je třeba zajistit rychlé a spolehlivé internetové připojení a vyloučit přístup neoprávněných uživatelů.

Existuje hned několik způsobů, jak data embedovat. Mezi nejpopulárnější a nejrozšířenější způsoby patří HTML iframe, webová komponenta či React SDK s voláním API [1].

2.1.1 Iframe

Nejjednodušší a nejrychlejší metodou embedování je využití použití iframu [1]. Iframe (inline frame) je HTML prvek, který umožňuje načíst HTML stránku uvnitř dokumentu [6]. Používá se pro vložení určitého obsahu z jednoho zdroje do druhého. To se poté jeví jako nové okno v dané stránce, jedná se ovšem o embedovaný iframe. Tuto možnost embedování využívají např. Google Mapy nebo YouTube [6]. Velkou výhodou je, že není třeba instalovat pro běh žádné dependence.

[1] Widgety nemusí být obecně pouze vizualizace, může se jednat také o ovládací prvky (např. výběrové seznamy, check boxy, textfieldy apod.). Tato možnost se využívá zejména ve vývojářských dashboardech pro rychlejší tvorbu vizualizací. V samotném produktu firmy se ale používá kvůli uživatelské přívětivosti pouze pro vizualizace [3].

2.1.2 Webová komponenta

Pokročilejší technikou pro embedování je použití webové komponenty. Výhoda opět spočívá v tom, že není třeba instalovat závislosti pro běh. Embedování probíhá pomocí knihovny, která se naimportuje přes HTML tag `<script>`. Poté je možno embedovat vizualizace pouze prostřednictvím naimplementovaných webových komponent [7].

2.1.3 React SDK + API

Poslední zmiňovanou možností je embedování prostřednictvím React SDK s voláním API.

SDK (Software development kit) je označení sady nástrojů pro tvorbu softwaru [8], které umožňuje vývojářům vytvářet rychleji a standartizovaně nové aplikace. API (Application Programming Interface) je rozhraní usnadňující komunikaci mezi dvěmi platformami. Uživatel specifikuje požadavek na data, rozhraní API provede volání na webový server, který následně tento požadavek zpracuje a odešle odpověď na specifikované volání (může se jednat např. o data ve formátu JSON). Díky tomu se krátí vývojový cyklus (automatizace), efektivně se poskytují nové služby uživatelům a může zlepšit reputaci důvěryhodnost značky.

K embedování je potřeba následovat instrukce (např. nainstalování dependencí, zprovoznění backendu apod.) a zajistit, že všechny kroky v nich obsažené budou splněny [9]. Jedná se tedy o programátorsky náročnější metodu, ovšem výhoda spočívá ve větší flexibilitě vývojářů, kteří mohou vizualizace snadno modifikovat [1].

2.2 Zabezpečení

Zabezpečení API je důležitou součástí v moderním inženýrství zajišťující bezpečnou a důvěryhodnou komunikaci mezi různými aplikacemi. S rostoucím významem API pro výměnu dat mezi systémy je nezbytné věnovat zvláštní pozornost implementaci efektivních bezpečnostních opatření. API authentication (autentizace) je řešením pro ověřování uživatelů - to umožňuje vlastníkovi daného API ochranu před neoprávněným přístupem ze strany uživatelů, kteří nemohou ověřit svou totožnost [10].

Řešení autentizace jsou obvykle nastavena tak, aby zablokovala přístup do API, pokud se při volání zjistí něco nesprávného či nevalidního s uživatelem. S tím se pojí druhá bezpečnostní složka, kterou je autorizace (authorization). Zatímco autentizace ověřuje identitu uživatele, autorizace se zabývá tím, jaké akce má daný uživatel povoleny [11]. Mezi nejpoužívanější metody patří HTTP authentication, API keys a OAuth 2.0.

2.2.1 HTTP autentizace

HTTP autentizace omezuje přístup k serveru pomocí předdefinovaných schémat.

Jedním z používaných schémat je Basic HTTP [12]. Uživatel, který chce ověřit svou identitu, tak může učinit vložením hlavičky Authorization s bezpečnostními údaji - nejčasteji tím bývá uživatelské jméno a heslo. Samotné ověřování pomocí HTTP Basic se ovšem nedoporučuje. Přenášené údaje jsou sice zakódované, ale nejsou zašifrované [12]. Proto je třeba v případě použití zajistit šifrované spojení (HTTPS/TLS). Atribut hlavičky se zakódovaným uživatelským jménem a heslem může vypadat např. jako na ukázce 2.1.

Zdrojový kód 2.1: Autorizační atribut - Basic schéma

```
1 Authorization: Basic drgnpdrgud653==,
```

Bearer schéma obsahuje tzv. bearer tokeny. Bearer token specifikuje, k jakým zdrojům může uživatel v API přistupovat [11]. Token je obvykle řetězec vygenerovaný serverem v reakci na požadavek na přihlášení. Opět je nutné zajistit zabezpečené spojení jako v případě Basic schématu. Atribut hlavičky s tokenem může vypadat např. jako na ukázce 2.2

Zdrojový kód 2.2: Autorizační atribut - Bearer schéma

```
1 Authorization: Bearer se5Edg345dsBNN-7df,
```

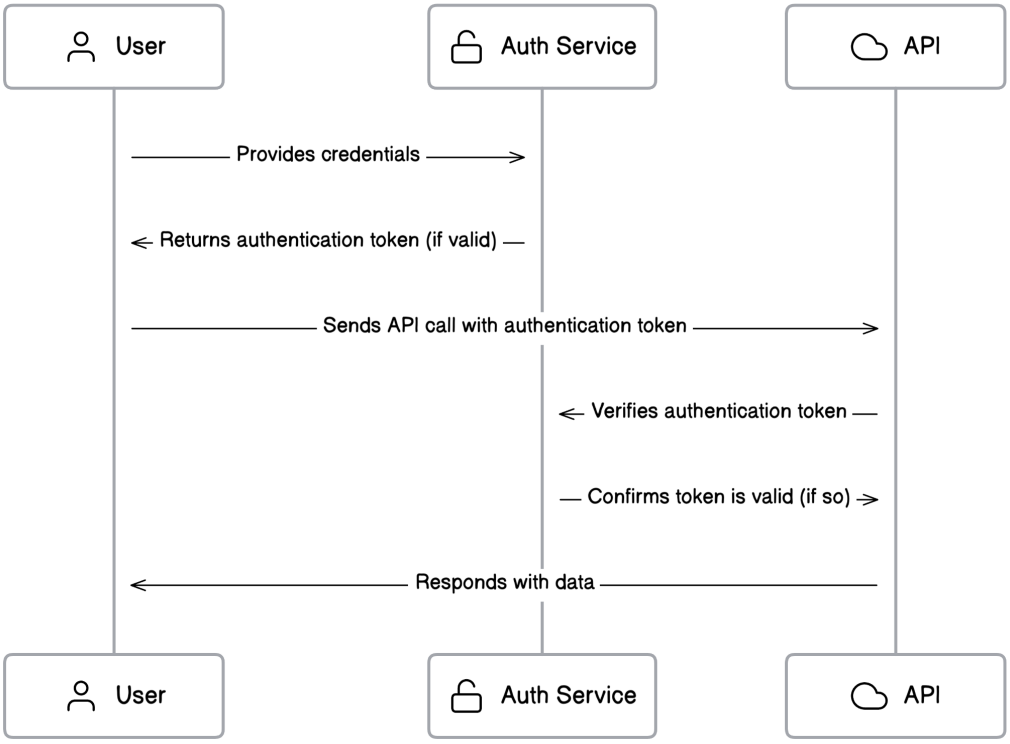
2.2.2 API key

API klíč (key) je jedinečný alfanumerický řetězec, který slouží k jednoznačné identifikaci klientů/aplikací [13], jenž API využívají. Narozdíl od tokenů nejsou časově omezené - to může znamenat bezpečnostní problém v případě, že při zadávání požadavku API by přenášený klíč mohl být zachycen [11] a s ním i celá síť, pokud by obsahovala jeden nezabezpečený bod. Proto je i v tomto případě nutné zajistit šifrované spojení. Pokud API bude sloužit pouze ke čtení, je zabezpečení API klíčem tou nejlepší volbou - uživatelé budou moci data stahovat, ale nikterak modifikovat či mazat [11].

2.2.3 OAuth

OAuth (Open Authorization) protokol byl vyvinut jako řešení pro udělování přístupu na předem definovanou dobu bez sdílené uživatelských jmen a hesel [14]. OAuth 2.0 je nejlepší volbou pro identifikaci osobních uživatelských účtů a udělování patřičných oprávnění [11]. Proces autentizace je vidět na obrázku 2.2.

Nejprve se uživatel přihlásí do systému - nejčastěji formou uživatelského jména a hesla, nebo klientského id a secretu. Tím se pošle HTTP požadavek na endpoint.



Obrázek 2.2: Autentizační diagram protokolu OAuth 2.0 [15]

Autorizace uživateli následně vrátí token. Pokud má uživatel již platný token, může přistupovat k datům. Naopak pokud mu token již expiroval nebo není platný, volání do API mu vrátí chybové hlášení.

Součástí autorizační odpovědi bývá i několik dalších údajů. Odpověď může vypadat např. jako na 2.3.

Zdrojový kód 2.3: Ukázková odpověď autorizačního serveru

```
{
        "access_token": "G7k4Q2s1bD5z9x",
        "token_type": "Bearer",
        "expires_in": 3600,
        "refresh_token": "5aB3c9Cx8P5y2z",
        "scope": "create"
}
```

Odpověď bývá ve formátu JSON (JavaScript Object Notation) a obsahuje informace o daném přístupovém tokenu (access_token). Kromě přístupového tokenu obvykle obsahuje i refresh_token. Ten umožňuje uživateli používat stále jeden token bez nutnosti generování nového [16]. Dále může obsahovat specifikující práva uži-

vatele (scope), platnost tokenu (expires_in) či typ tokenu (token_type), jež některé z nich byly zmíněny v předchozí kapitole. Přesný formát odpovědi je ale specifikován samotným API - tudíž každé API může mít jiné délky přístupových a obnovovacích tokenů, časové platnosti tokenů či i samotné atributy v odpovědi a jejich názvy.

2.3 Existující řešení

Embedded analytics problematikou se zabývá mnoho firem a každá z nich disponuje jinými způsoby řešení. Zde budou zmíněny firmy, které se objevují často na vrchních příčkách internetových recenzí či firmy, které byly pro inspiraci doporučeny externím zadavatelem. Pro aktuálnost jsou brány informace ze zdrojů z roků 2022/2023.

2.3.1 GoodData

Firma GoodData byla zmíněna zadavatelem jako jedna z nejvýraznějších na trhu. Dodává embedded analytics systémy firmám jako je VISA, Zalando či Bentley [17]. Embedování provádí třemi zmíněnými způsoby - HTML iframem, webovou komponentou či React SDK. Zákazníkům nabízí přizpůsobení dashboardů tak, aby odpovídaly zákazníkovo značce. GoodData získaly v roce 2023 hned několik ocenění firmou TrustRadius, důvěryhodnou platformou v oblasti B2B (bussines to bussines) [18]. Firma GoodData obdržela ocenění Best Value for Price, Best Relationship či Top Rated 2023. Zákazníci firmy GoodData si v recenzích chválí hlavně snadné používání, zákaznickou podporu, širokou škálu podporovaných databází či uživatelsky přívětivé rozhraní [18]. Naopak se zmiňují, že dokumentace může být nedostačující či některé endpointy v API mají neintuitivní parametry. I tak ale 92% recenzentů uvádí, že by si produkt zakoupili znovu.

2.3.2 Microsoft Power BI

Mezi často zmiňovaným produktem je Microsoft Power BI, který je ve svém oboru nejlepší v oblasti bezpečnosti a zabezpečení [19], jelikož Microsoft zaměstnává více než 3500 bezpečnostních expertů. V produktu nabízí také tzv. playground, který umožňuje uživatelům testovat funkce a vlastnosti daných dashboardů před úplným nasazením. Je také podporvána integrace s cloudovou službou Microsoft Azure. Na webu TrustRadius je k Microsoft Power BI podstatně méně recenzí než k produktu GoodData (5x méně). Uživatelé zmiňují dlouhou dobu při zpracování většího objemu dat či vysokou provozní cenu [20].

2.3.3 Powered by Looker

V oblasti modelování data vyniká platforma Powered by Looker [19]. Jedná se o produkt služby Google Cloud, který poskytuje způsob, jak sledovat změny a prohlížet jejich historii v databázi. Disponuje modelovacím jazykem LookML založeným na SQL, který slouží k vytváření sémantických datových modelů. Pomocí něj lze popisovat jednotlivé dimenze, agregáty, výpočty či datové vztahy v databázi [21]. Z toho vyplývá, že datové modely jsou rozšiřitelné, opakovatelně použitelné a konzistentní a tudíž efektivní. V roce 2022 byl tento produkt oceněn Top Rated a Most Loved společností TrustRadius [22]. Uživatelé zmiňují, že produkt může být pomalejší a že nástroje na přizpůsobení dashboardů by mohly být vylepšeny a rozšířeny.

2.3.4 Tableau

Tableau vyniká ve vytváření estetických a interaktivních vizualizací [23]. Dále nabízí zákazníkům publikaci vizualizací prostřednictvím produktu Tableau Online. Cena produktu začíná na $70 měsíčně za jednoho uživatele [24]. Zákazníci produkt hodnotí velmi kladně - 100% z nich uvádí, že implementace proběhla jak očekávali, 98% jsou s výsledkem spokojeni a 91% by si tento produkt zakoupilo zase [24]. Silné stránky tohoto produktu spočívají v automatickém generování kódu pro embedování a jednoduché vkládání dashboardů do webových stránek [23]. Kritika naopak zaznívá na zákaznickou podporu či pomalé načítání při velkém množství dat [24].

Návrh knihovny a uživatelského rozhraní

3

V této kapitole bude popsán detailní návrh knihovny, uživatelského rozhraní a technologie, jež budou použité k realizaci.

3.1 Technologie

3.1.1 JavaScript a React

Pro vývoj knihovny bude využit JavaScript s knihovnou React. Důvodem zvolení těchto technologií je požadavek ze strany zadavatele. React a JavaScript jsou dnes široce používané technologie ve vývoji webových aplikací.

React je snadný na naučení, obsahuje málo konceptů, které je třeba se naučit [25]. Jeho instalace je snadná - stačí pouze v kódu nainstalovat a naimportovat jeho knihovnu. Využívá speciální JSX syntaxe, která sice vypadá jako HTML, ale ve skutečnosti je tato syntaxe převáděna na HTML. Jelikož je React hojně využíván v aplikaci Facebook či na Instagramu, dostává se mu velké podpory právě i z tohoto odvětví - čtyři největší přispěvatelé knihovny React jsou zaměstnanci Facebooku [25]. Kromě Facebooku využívají React značky jako jsou např. Netflix, airbnb, BBC News či PayPal [26]. Díky virtualizaci a uchováváním DOM poskytuje React velmi rychlé vykreslování, přičemž všechny změny se snadno promítají do virtuálního DOM.

DOM (Document Object Model) je strukturovaná reprezentace jazyka HTML, které reprezentuje celé uživatelské rozhraní jako stromovou datovou strukturu [27]. Každý prvek uživatelského rozhraní tvoří v DOM stromu právě jeden uzel. Dojde-li ke změně uživatelského rozhraní, DOM se aktualizuje a při každé změně se vykresluje znovu, což výrazně ovlivňuje výkon aplikace. Toto lze vyřešit použitím virtuálního DOM. Při přidávání nových věcí do aplikace se vytvoří virtuální DOM, která je reprezentován jako strom. Novější virtuální DOM se porovnává se star-

ším, aby zaznamenal změny. Poté zjistí, jak je možné tyto změny provést pomocí skutečného DOM a aktualizované prvky se následně vykreslí.

Na popularitě Reactu přidává také použití knihovny Redux, která umožňuje uchování dat jako jeden objekt (stav). K tomuto objektu se dá pak jednoduše přistupovat, což značně usnadňuje práci s daty. Je-li tento stav změněn, aplikace se překreslí a zobrazení se stále synchronizuje se souvisejícími daty [26].

React se kvůli těmto vlastnostem doporučuje využívat u:

1. Obsáhlých uživatelských rozhraní
2. Rozsáhlých aplikací
3. Aplikací náročné na výkon
4. Multi-platformních aplikací

Na frontend komponent knihovny a rozhraní pro obnovu tokenů bude použita knihovna MUI, která nabízí širokou škálu předdefinovaných komponent a stylů [28].

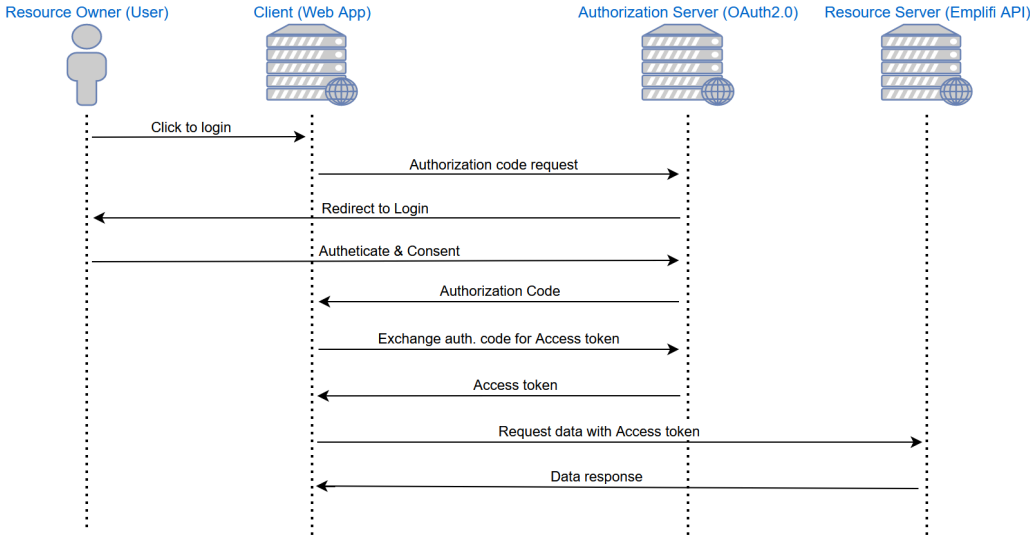
3.1.2 Emplifi API

Data potřebná k embedování grafů budou získávána prostřednictvím Emplifi Public API. Toto rozhraní poskytuje klientům snadný a oblíbený způsob, jak získat přístup k potřebným datům. Emplifi API se nachází na veřejné webové adrese a žádat o data může každý, kdo má přístupový token. Struktura dotazů a ukázkové dotazy jsou dostupné na veřejné dokumentaci [29].

3.1.2.1 Vytvoření tokenu

Vytvoření tokenu je umožněno pouze osobám, jež mají u firmy Emplifi vytvořený účet v produktu Suite. Na diagramu 3.1 je schéma, které udává průběh akcí během žádání o přístupový token a zaslání prvního requestu s žádostí o data.

Pro vytvoření tokenu je třeba uživatele přesměrovat na webovou URL `https://api.emplifi.io/oauth2/0/auth`, kde bude požádán o udělení souhlasu z jeho Emplifi Suite účtu. Do URL, na kterou uživatele přesměrujeme, bude třeba vložit několik parametrů potřebných k autentizaci a k následnému vrácení tokenu.



Obrázek 3.1: Diagram vytvoření Emplifi API tokenu [29]

Tabulka 3.1: Seznam všech parametrů vkládaných do URL adresy při tvorbě tokenu [29]

`client_id`	Jedinečný klientský identifikátor pro aplikaci.
`redirect_uri`	Návratová adresa. Na tuto adresu bude při úspěšné autorizaci přesměrován přístupový token. Zároveň tato adresa musí být povolena vývojáři v interní databázi.
`response_type`	Typ - Emplifi momentálně podporuje pouze hodnotu "code".
`scope`	Specifikuje prostředky oddělené mezerou, ke kterým aplikace může přistupovat klientským účtem. Ty při udělování souhlasu budou předloženy ke schválení.
`state`	Hodnota sloužící k validaci přijaté odpovědi. Při zaslání requestu je vložena hodnota do parametru, autorizační server jí při zpracování dotazu odešle zpět s daty. Uživatelská aplikace by následně měla porovnávat, zda tato zaslaná hodnota je shodná s původní - to slouží k zabránění útoků Cross-site Request Forgery.
`prompt`	Musí být nastavena na hodnotu "consent", pokud má být refresh token vrácen společně s přístupovým.

Pokud veškeré parametry budou správně nastaveny a uživatel udělí souhlas s používáním Suite účtu, na URL "redirect_uri"se vrátí zpráva s autorizačním kódem. Tento kód slouží k výměně za přístupový token. O ten je možno si zažádat POST requestem na adresu `https://api.emplifi.io/oauth2/0/token`. Struktura requestu je popsána v tabulkách 3.2 3.3.

Tabulka 3.2: Hlavička requestu na token [29].

`Authorization`	Řetězec "Basic"s klientským identifikátorem a zakódovaným klientským secretem oddělený dvojtečkou v Base64. Výsledná hodnota vypadá např. takto: "Basic Y2xpZW50X2lkOmNsaWVudF9zZWNyZXQ=."
`Content-Type`	Hodnota "application/x-www-form-urlencoded"

Tabulka 3.3: Tělo requestu na token [29].

`code`	Řetězec sloužící k následné validaci při zpětném volání.
`grant_type`	Hodnota "authorization_code"
`redirect_uri`	URL adresa, kam bude přesměrován token. Musí být stejná jako v tabulce 3.1

Následně obdržíme token ve formátu jako na 2.3. Přesné návratové hodnoty jsou popsány v tabulce 3.4.

Tabulka 3.4: Tělo odpovědi se zaslaným tokenem [29].

`access_token`	Přístupový token k Emplifi Public API.
`token_type`	Jediný podporovaný typem je "bearer".
`expires_in`	Platnost tokenu ve vteřinách.

(tabulka pokračuje na další stránce)

Tabulka 3.4: Tělo odpovědi se zaslaným tokenem [29].

`refresh_token`	Token pro obnovení přístupového tokenu. Zaslán pouze v případě, pokud `scope` obsahuje řetězec "offline_access".
`scope`	Prostředky, které byly uživatelem uděleny. Odděleny mezerou.

Užívání tokenů má ovšem nastavené limity - 500 requestů za hodinu pro uživatele, 1000 requestů za hodinu pro účet [29]. Pokud bude tato hranice přesažena, API vrátí odpověď s informací, že uživatel či účet překročili maximální limit dotazů.

3.1.2.2 Endpointy

Má-li uživatel platný token, může začít se zasíláním dotazů. Emplifi API disponuje množstvím endpointů, přičemž endpointy využívané k embedování jsou zmíněné v tabulce 3.5.

Tabulka 3.5: Endpointy používané k embedování.

`/3/omni/metrics`	Získávání dat pro vizualizace. Data jsou ve formátu omni.
`/3/omni-studio`	Získávání widget konfigurací, fieldů a vytváření Omni API tokenů
`/oauth2/0`	Vytváření Public API tokenů.

Endpoint omni-studio bylo nutné vytvořit zadavatelem, jelikož Omni API endpointy bylo možné volat pouze při použití firemní VPN. Nyní lze tyto endpointy provolávat prostřednictvím Public API. Původní strukturu Omni API requestu lze vložit do těla public API requestu, což umožní uživateli embedovat grafy i mimo VPN.

3.1.2.3 Knihovny pro fetchování dat

V současné době existuje mnoho knihoven určené pro fetchování dat. Data bude nejprve nutné načíst na backend běžící aplikace, která je následně vrátí na frontend.

Na backendu probíhá fetchování dvojím způsobem - využitím zabudovaného fetch API v Node.js a knihovnou axios. Pro načítání dat z backendu na frontend budeme využívat rozhraní JavaScriptu Fetch API a knihovnu TanStack React Query. Její výhodou je cachování requestů [30] - pokud byl poslán jednou již stejný dotaz na data, načtení dat neproběhne z API, ale z uložené cache. To omezí počet requestů zaslaných na API a zmenší prodlevu vizualizace (data načtou rychleji z cache než z API).

3.1.3 PreJSON a Vision

Interní knihovny, které budou využiti k upravování widget konfigurací (PreJSON) a následné vizualizaci grafů (Vision).

PreJSON slouží k expandování konfigurací widgetů. Stažené konfigurace widgetů obsahují tzv. preJSON hodnoty, které jsou později dodefinovány uživatelem. Např. je-li stažena konfigurace, která obsahuje nespecifikovaný parametr, můžeme jej dodefinovat použitím instance PreJSONu [31]. Na zjedodušené ukázce 3.1 je zobrazen PreJSON objekt, který by mohl specifikovat tělo dotazu.

Zdrojový kód 3.1: Neexpandovaný PreJSON objekt

```
{
        customer_id: 12651141417427,
        time: "P30D/now[sD]",
        platform: ${string:platform_name}
}
```

Atributy `customer_id` a `time` jsou již předdefinové, ale na uživateli zůstává možnost volby parametru `platform`. Datový typ parametru musí být vždy specifikován. Zde se očekává string hodnota, která může nabývat (opět pouze pro ukázku) např. hodnot "instragram", "facebook", "snapchat", "linkedin"apod. Uživatel tedy zvolí jednu ze sociálních sítí a pomocí funkce `expand()` může tuto hodnotu dodefinovat. Výsledek může vypadat poté jako na ukázce 3.2.

Zdrojový kód 3.2: Expandovaný PreJSON objekt

```
{
        customer_id: 12651141417427,
        time: "P30D/now[sD]",
        platform: "snapchat"
}
```

Knihovna Vision slouží k vytváření vizualizací. Obsahuje komponentu, která ovšem nespecifikuje, jak daná vizualizace bude vypadat [32], ale slouží k vykreslení vizualizace. Veškeré data a konfigy vizualizace (např. jakou barvu bude mít graf, jaká

je popsána osa X apod.) jsou předány uživatelem a knihovna se pouze postará o vykreslení. Tato výsledná vizualizace neobsahuje ale např. nadpisy, nápovědy apod., to bude zajišťovat výsledná knihovna této bakalářské práce.

O vykreslování se stará komponenta <Vision/>. Pro potřeby BP budou využívány parametry uvedené v tabulce 3.6.

Tabulka 3.6: Parametry komponenty <Vision/>

spec	Validní JSON expandovaná konfigurace grafu (druh grafu, popisky os, barvy grafu apod.).
input	Data zobrazována v grafu.

Následně je třeba tuto komponentu s předanými parametry vložit do komponenty <VisionContextProvider/>, aby došlo ke správnému vyrenderování.

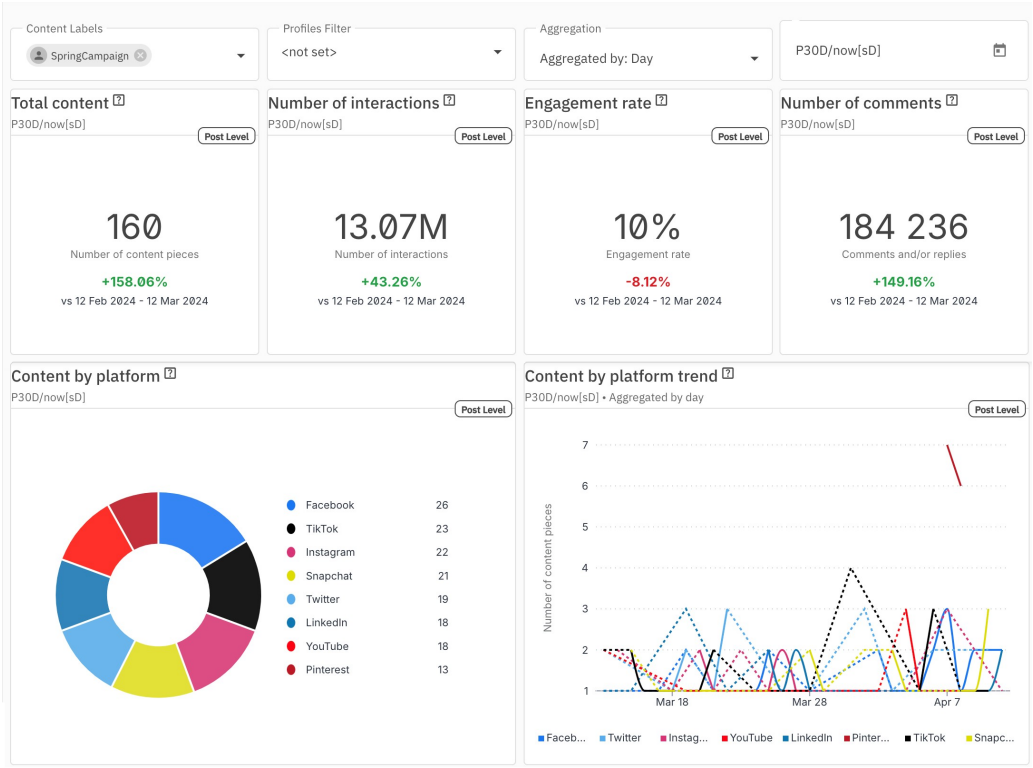
3.2 Návrh knihovny

Knihovna se bude skládat jak z front-endové části (vykreslování widgetů), tak i back-endové (provolávání API endpointů). Jejím cílem bude identifikovat, jaká data stáhnout z API a patřičně je zpracovat, případně reagovat na chyby (nevalidní data, vypršení platnosti tokenu).

3.2.1 Omni Studio

Veškeré vizualizace se nachází v interní aplikaci Omni Studio, která slouží vývojářům k rychlému a jednoduchému zobrazení grafů. K embedování bude zapotřebí mít přístup do této aplikace (přes VPN). Tato interní aplikace obsahuje stovky dashboardů, na nichž se nachází widgety. Jak již bylo zmíněno, widgety nejsou ovšem jen vizualizované grafy, ale může se jednat např. o tabulky, výběrové seznamy či prostý text sloužící k popisu boardu. Knihovna má sloužit k embedování vizualizací (tudíž pouze Vision komponent - jak se v aplikaci grafy nazývají), ale bude navržena tak, aby v budoucnu byla rozšiřitelná a šla použít i pro jiné druhy widgetů.

Ukázkový dashboard z Omni Studia je znázorněn na obrázku 3.2. Vidíme, že dasboard obsahuje nejen vizualizace, ale i ovládací prvky. Každý tento widget má svoje jedinečné ID, které je dostupné, pokud se rozklikne nastavení widgetu. Zároveň Omni Studio umožňuje uživateli zobrazit konfiguraci grafu (neexpandované), datové requesty (neexpandované) a i samotná data. V jedné ze sekcí widgetu jsou i tzv. debuggovací nástroje, ve kterých se nachází pole JSON parametrů, které je



Obrázek 3.2: Dashboard v Omni Studiu

dodefinovat, aby graf mohl být vykreslen. Na obrázku 3.2 je zřejmé, že uživatel pro realizaci vykreslení musel vybrat hodnoty v ovládacích prvkách (konkrétně časovou škálu, časovou agregaci a hodnotu labelu - hodnota pro filtry zůstala nevyplněná, jelikož je nepovinná - udává pouze omezení na datový request, pokud by chtěl uživatel data ještě nějak dál extrahovat - např. filtrování na základě sociální sítě, typu interakcí apod.).

Tyto hodnoty se následně převedou do pole JSON objektů a následně se při vizualizaci uplatní. Tím grafy zůstávají dynamické, protože při každé uživatelské interakci se ihned překreslí s novými daty.

Tyto parametry budou klíčové pro správnou funkcionalitu knihovny - zajistí správné vykreslení a pokud nebudou v rámci aplikace specifikovány nějakými prvky (např. že by uživatelova aplikace obsahovala inputy/výběrové seznamy pro všechny nespecifikované hodnoty), bude nutné je specifikovat ručně. To bude provedeno předáním přes properties React komponenty.

3.2.2 Funkcionalita knihovny

Knihovna tedy bude obsahovat obecnou komponentu `<Widget/>`, která bude reprezentovat právě jednu vizualizaci. Bude nutné komponentě předat číslo boardu,

číslo widgetu a případně parametry, které budou expandovány knihovnou PreJSON v konfiguraci widgetu. Tato komponenta se postará o veškeré fetchování dat a následnou vizualizaci. Jelikož Vision knihovna vizualizuje pouze samotný graf, bude nutné, aby Widget komopnenta uměla zobrazit i nadpisy, podnadpisy a další prvky obsažené v hlavičce. Bude tedy třeba zajistit základní stylování (např. fonty a velikost písma), ale uživateli bude umožněno vložení CSS stylů opět přes propsu komponenty - to nabídne uživateli větší možnosti, jak si embedovaný graf sám vystylovat. Postup při vizualizaci bude vypadat přibližně takto:

1. Uživatel zadá ID boardu a widgetu + další potřebné parametry
2. Komponenta provede veškerá potřebná volání - stáhne konfigurace, proběhne expandace konfigů apod.
3. Aplikují se předané CSS styly
4. Výsledné zobrazení grafu

Výsledná knihovna bude přístupná přes npm (Node Package Manager), což zajistí její snadné stažení a použití. Instalace bude vyžadovat pouze nainstalovaného správce balíčků npm a použití budete detailně vysvětleno v uživatelské dokumentaci.

3.3 Návrh UI

Při navrhování uživatelského rozhraní je třeba myslet na UX (User Experience). Výsledné rozhraní musí být přehledné, uživatelsky přívětivé a nezasekávat se. Pro lepší UX bude rozhraní obsahovat nejen sekci pro správu tokenů, ale také sekci pro náhled embedovaných grafů a následné generování zdrojového kódu pro tyto náhledy. Tím se zvýší použitelnost tohoto UI, jelikož věci potřebné k embedování sloučí do jednoho UI (tj. bude se jednat o jednostránkovou aplikaci). Ze strany zadavatele je důraz především na funkcionalitu a UX, nikoliv na estetiku. Proto toto bude zohledněno při navrhování.

3.3.1 Vytváření tokenů

K vytvoření tokenů bude sloužit patička stránky. V ní se budou nacházet dvě tlačítka - jedno pro vytvoření Public API tokenu, druhé pro Omni API token. K vytvoření těchto tokenů je třeba být přihlášen v účtu Suite, jak bylo dříve zmíněno. Po stisknutí tlačítka bude uživatel přesměrován na stránku, kde udělí patřičné oprávnění, že se pro jeho účet vytvoří token. Po potvrzení se uživateli token uloží do local storage webového prohlížeče. To umožní okamžitou možnost embedování v UI a zároveň

snadnou přístupnost k tomuto tokenu. Pokud bude chtít uživatel token používat ve své aplikaci, jednoduše jej z local storage zkopíruje k sobě do aplikace, kde jej bude dále moci normálně využívat. To by platilo pro případ, kdy by jeden z tokenů vypršel. Ovšem bude-li uživatel nový a bude nutno vygenerovat oba dva tokeny pro potřebné vizualizace, bude přidáno tlačítko, které uživateli zkopíruje oba dva tokeny ve formátu .env souboru. Uživatel tedy nebude muset sám hodnoty přepisovat, ale pouze je vloží zkopírované do envu. Zkopírované tokeny budou ve formátu:

```
ACCESS_TOKEN=pis4185dfgfdesfDs5asd
OMNI_API_TOKEN=s5Z9sB=Fd--lK67a44ed12
```

3.3.2 Náhled vizualizací

Součástí uživatelského rozhraní bude také sekce pro náhled na vizualizované grafy. Ten bude obsahovat základní uživatelské vstupy (všechny, které bude přijímat výsledná komponenta z naimplementované knihovny), díky kterým se vizualizace budou moci rychle ovládat a vykreslovat. Pro rychlé embedování bude vytvořeno i tlačítko, které otevře uživateli dialog, v němž bude vygenerovaný zdrojový kód, kterým by se v jeho aplikaci daný graf vykreslil. Výstup bude vypadat zhruba takto:

```
<Widget widgetID={1598} boardID={12}
params={{time:"now"}} width={500}/>
```

3.4 Dokumentace

Výstupem bude i podrobná dokumentace - a to jak uživatelská, která bude specifikovat použití pro koncové uživatele, tak i programátorská, která bude obsahovat souhrn informací potřebné pro další vývoj. Dokumentace bude obsahovat následující body:

1. Instalaci knihovny a její následnou integraci do uživatelovi aplikace
2. Stručný návod k používání uživatelského rozhraní pro obnovu tokenů
3. Sekci pro vývojáře

Přístupu k dokumentaci je několik - od stručného souboru .pdf až po samostatnou webovou aplikaci. Aby dokumentace byla přehledná a snadno rozšiřitelná, bude dokumentace statická stránka přístupná na veřejné URL adrese. Pro tuto možnost využijeme Docosaurus - generátor statických stránek určen pro generaci právě uživatelských dokumentací.

Docosaurus umožní ze souborů s příponou .md (soubory běžně používané pro dokumentace v gitových repozitářích) zbuildit statickou webovou stránku, která uživateli poskytne přehledné rozhraní. Výsledná dokumentace bude vydána na veřejné internetové stránce, aby k ní byl snadný přístup odkudkoliv.

Implementace 4

Tato kapitola popisuje implementaci knihovny a uživatelského rozhraní.

4.1 Implementace knihovny

4.1.1

4.2 Implementace uživatelského rozhraní

Závěr 5

Aktuálně je praktická část bakalářské práce ve formě, kdy React aplikace dokáže vizualizovat grafy. Grafy lze vizualizovat pomocí playgroundu, kterému bude v budoucnu ještě upraven frontend. Zároveň aplikace dokáže obnovovat OAuth2 tokeny (jak token pro Emplifi Public Api, který vrací schémata widgetů, tak i token pro Omni API, který vrací data do widgetů). Toto je udělané pouze formou dvou tlačítek. Ty jsou automaticky vypnuté, pokud jsou tokeny validní - momentálně se uchovávají v local storage webového prohlížeče. Externí zadavatel určí co dále. Jsou i vypisovány základní stavy tokenu (platný, neplatný, neexistující) a vizualizace v playground nelze embedovat, dokud tyto tokeny nejsou platné. Playground umožňuje uživatelům widgety i upravovat - například zadáním CSS stylu či zadáním šířky/-výšky. Pro debuggování a kontrolu jsou umožněny i výpisy schémat daného widgetu - ty se ovládají jednoduchým přepínačem. Jsou ošetřeny základní chyby, které mohou nastat při vizualizaci (chybějící tokeny, nezadané parametry pro PreJSON apod.).

Nyní bych rád pokračoval v zefektivnění kódu, dopsáním dokumentačních komentářů, napsáním jednotkových testů pro kritické sekce programu, zlepšení frontendu pro playground a pro rozhraní pro obnovu OAuth 2.0 tokenů. Zároveň bude třeba ošetřit všechny výjimky, které mohou nastat. Dalším krokem bude převést React aplikaci na knihovnu a domluvit se se zadavatelem, jak pokračovat dále - například zda-li stačí vizualizovat jednotlivé widgety, nebo celé dashboardy. Jako poslední budou provedeny i funkcionální testy podle scénářů, které budou přiloženy ve finálním odevzdání bakalářské práce.

Bibliografie

1. SEIDELOVA, Tereza. *Embedded Analytics: All You Need To Know*. GoodData, 2023-10-31. Dostupné také z: https://www.gooddata.com/blog/what-embedded-analytics/.
2. BRUK, Vojtěch. *Co je Dashboard?* 2023-02-22. Dostupné také z: https://vojtechbruk.cz/pojem/dashboard/.
3. *Dashboard*. Emplifi, 2023-12. Dostupné také z: https://docs.emplifi.io/platform/latest/home/dashboard. [Online: accessed 9.Dec.2023].
4. Emplifi, 2024-04. Dostupné také z: https://emplifi.io/resources/blog/linkedin-insights-emplifi-analytics-dashboard-widgets. [Online: accessed 11.Apr.2024].
5. *Software as a Service (SaaS): Definition and Examples*. Investopedia, 2022-12-15. Dostupné také z: https://www.investopedia.com/terms/s/software-as-a-service-saas.asp.
6. CHINN, Kerrie-Anne. *What's an iFrame and how can I use it?* 2017-09-14. Dostupné také z: https://www.go1.com/blog/whats-iframe-can-use. [Online: accessed 7.Apr.2024].
7. *Embed Visualizations Using Web Components*. GoodData. Dostupné také z: https://www.gooddata.com/docs/cloud/embed-visualizations/web-components/. [Online: accessed 7.Apr.2024].
8. *SDK vs. API: What's the Difference?* IBM, 2021-07-13. Dostupné také z: https://www.ibm.com/blog/sdk-vs-api/.
9. *Embed Visualizations Using React SDK*. GoodData. Dostupné také z: https://www.gooddata.com/docs/cloud/embed-visualizations/react-sdk/. [Online: accessed 7.Apr.2024].
10. *What is API Authentication?* Noname Security, 2023-06-08. Dostupné také z: https://nonamesecurity.com/learn/what-is-api-authentication/.

11. LEVIN, Guy. *4 Most Used REST API Authentication Methods*. 2019-07-26. Dostupné také z: https://blog.restcase.com/4-most-used-rest-api-authentication-methods/.
12. *HTTP authentication*. 2023-12. Dostupné také z: https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication. [Online: accessed 9.Dec.2023].
13. *What is an API key?* Amazon Web Services, 2023-12. Dostupné také z: https://aws.amazon.com/what-is/api-key/. [Online: accessed 9.Dec.2023].
14. STARREVELD, Albert. *Understanding OAuth2*. 2023-08-02. Dostupné také z: https://medium.com/web-security/understanding-oauth2-a50f29f0fbf7.
15. *Auth Flow Diagrams*. Eraser, 2024-04. Dostupné také z: https://www.eraser.io/use-case/auth-flow-diagrams. [Online: accessed 7.Apr.2024].
16. *Access Token Response*. OAuth, 2023-12. Dostupné také z: https://www.oauth.com/oauth2-servers/access-tokens/access-token-response/. [Online: accessed 9.Dec.2023].
17. *GoodData embedded analytics platform*. GoodData, 2023-12. Dostupné také z: https://www.gooddata.com/embedded-analytics/.
18. *What is GoodData?* TrustRadius, 2023-12. Dostupné také z: https://www.trustradius.com/products/gooddata/reviews?qs=pros-and-cons#reviews. [Online: accessed 9.Dec.2023].
19. PORRITT, Stephen. *Best Embedded Analytics Tools (2023)*. 2023-01-30. Dostupné také z: https://technologyadvice.com/blog/information-technology/embedded-analytics-software/.
20. *What is Microsoft Power BI Embedded?* TrustRadius, 2023-12. Dostupné také z: https://www.trustradius.com/products/microsoft-power-bi-embedded/reviews?qs=pros-and-cons. [Online: accessed 9.Dec.2023].
21. *Introduction to LookML*. Google Cloud, 2023-12. Dostupné také z: https://cloud.google.com/looker/docs/what-is-lookml. [Online: accessed 9.Dec.2023].
22. *What is Looker?* TrustRadius, 2023-12. Dostupné také z: https://www.trustradius.com/products/looker/reviews. [Online: accessed 9.Dec.2023].
23. REHANA, Sharon. *Auth Flow Diagrams*. Comparing Embedded Analytics Solutions in 5 Business Intelligence (BI) Tools, 2024-04. Dostupné také z: https://www.analytics8.com/blog/comparing-embedded-analytics-solutions-in-five-business-intelligence-tools/#tableau.
24. *What is Tableau Desktop?* TrustRadius, 2024-04. Dostupné také z: https://www.trustradius.com/products/tableau-desktop/reviews. [Online: accessed 8.Apr.2024].

25. SURVE, Suraj. *Why You Should Use React.js For Web Development*. 2021-02-18. Dostupné také z: `https://www.freecodecamp.org/news/why-use-react-for-web-development/`.

26. HUTSULYAK, Oleksandr. *10 Key Reasons Why You Should Use React for Web Development*. 2023-03-12. Dostupné také z: `https://www.techmagic.co/blog/why-we-use-react-js-in-the-development/`.

27. *What is Dom in React?* Javatpoint, 2023-12. Dostupné také z: `https://www.javatpoint.com/what-is-dom-in-react`. [Online: accessed 9.Dec.2023].

28. *MUI documentation*. 2024-04. Dostupné také z: `https://mui.com/`. [Online: accessed 11.Apr.2024].

29. *Emplifi API Documentation*. Emplifi, 2024-04. Dostupné také z: `https://api.emplifi.io/`. [Online: accessed 9.Apr.2024].

30. *Powerful asynchronous state management for TS/JS, React, Solid, Vue, Svelte and Angular*. 2024-04. Dostupné také z: `https://tanstack.com/query/latest`. [Online: accessed 11.Apr.2024].

31. KACEROVSKÝ, Michal. *PreJSON documentation*. Emplifi, 2024-04. [Attached in bachelor thesis .zip file as .pdf].

32. KACEROVSKÝ, Michal. *Vision documentation*. Emplifi, 2024-04. [Attached in bachelor thesis .zip file as .pdf].

Seznam obrázků

2.1 Ukázka dashboardu v produktu firmy Emplifi [4] 5
2.2 Autentizační diagram protokolu OAuth 2.0 [15] 9

3.1 Diagram vytvoření Emplifi API tokenu [29] 15
3.2 Dashboard v Omni Studiu 20

Seznam tabulek

3.1 Seznam všech parametrů vkládaných do URL adresy při tvorbě tokenu [29] 15
3.2 Hlavička requestu na token [29]. 16
3.3 Tělo requestu na token [29]. 16
3.4 Tělo odpovědi se zaslaným tokenem [29]. 16
3.4 Tělo odpovědi se zaslaným tokenem [29]. 17
3.5 Endpointy používané k embedování. 17
3.6 Parametry komponenty <Vision/> 19

Seznam výpisů

2.1 Autorizační atribut - Basic schéma 8
2.2 Autorizační atribut - Bearer schéma 8
2.3 Ukázková odpověď autorizačního serveru 9
3.1 Neexpandovaný PreJSON objekt 18
3.2 Expandovaný PreJSON objekt 18