VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

INFORMAČNÝ SYSTÉM PRE TVORBU GEOGRAFICKÝCH OBJEKTOV

INFORMATION SYSTEM FOR CREATING GEOGRAPHIC OBJECTS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **ŠIMON FEŇKO**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JIŘÍ HYNEK, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

148554

Ústav: Ústav informačních systémů (UIFS)
Student: **Feňko Šimon**
Program: Informační technologie
Specializace: Informační technologie

Název: **Informační systém pro správu geografických objektů**
Kategorie: Informační systémy
Akademický rok: 2022/23

Zadání:

1. Prostudujte problematiku zpracování a vizualizace geografických dat na webu a prozkoumejte existující technologie určené pro tento účel (např. Leaflet, d3-geo, Geovisto, apod.).
2. Prostudujte principy reprezentace a ukládání geografických objektů (např. formát GeoJSON). Prozkoumejte existující nástroj knihovny Geovisto určený pro tento účel a existující API pro automatizované stahování definic známých geografických objektů (např. státy, regiony, apod.).
3. Analyzujte požadavky na tvorbu geografických objektů exportovatelných ve formátu GeoJSON. Zaměřte se na uživatelskou přívětivost a usnadnění celého procesu tvorby.
4. Dle výsledků analýzy navrhněte informační systém pro správu geografických objektů vytvořených pomocí nástroje Geovisto.
5. Implementujte navržený informační systém.
6. Výsledné řešení otestujte a navrhněte další rozšíření.

Literatura:

- Hynek, J., Kachlík, J. a Rusňák, V.: Geovisto: *A Toolkit for Generic Geospatial Data Visualization*. In *VISIGRAPP (3: IVAPP)* (pp. 101-111).
- Tlčina, A.: *Webová aplikácia pre definíciu grafických objektov na mape*. Brno, 2021. Diplomová práca. Vysoké učení technické v Brně, Fakulta informačních technologií.
- Dent, B., D., a spol.: *Cartography: Thematic Map Design*. McGraw-Hill Higher Education 2009, ISBN 978-128-3388-023.
- Butler, et al. *The GeoJSON Format* [online]. Internet Engineering Task Force (IETF). 2016 [cit. 2021-10-09]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc7946
- Leaflet: *Leaflet API reference* [online]. 2021 [cit. 2022-10-15]. Dostupné z: https://leafletjs.com/reference-1.7.1.html

Při obhajobě semestrální části projektu je požadováno:
Body 1 až 4.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Hynek Jiří, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1.11.2022
Termín pro odevzdání: 31.7.2023
Datum schválení: 18.10.2022

Abstrakt

Cieľom tejto semestrálnej práce je navrhnúť a implementovať informačný systém pre správu geografických objektov, ktoré budú vytvorené pomocou programovej knižnice Geovisto. V rámci práce by mal byť uživateľ schopný vďaka nástrojom svoje vytvorené súbory vo formáte GeoJSON ukladať, zdieľať ich s inými uživateľmi a importovať vlastné. Systém bol navrhnutý a implementovaný v jazyku JavaScript a takisto za pomoci frontendovej knižnice JavaScriptu na vytváranie používateľských rozhraní React. Databáza uživateľov a ich uložených súborov typu GeoJSON bola implementovaná za pomoci objektovo-relačného databázového systému PostgreSQL.

Abstract

The aim of this semester project is to design and implement an information system for managing geographic objects created using the Geovisto programming library. As part of the project, the user should be able to save their created geoJSONs using tools, share them with other users, and import their own geoJSON files. The system was designed and implemented in JavaScript and utilizes the JavaScript frontend library React to create user interfaces. Additionally, the user database and their stored GeoJSONs were implemented using the object-relational database system PostgreSQL.

Kľúčové slová

geografické dáta, geografický informačný systém, Leaflet, webové frameworky, UI frameworky, užívateľské rozhranie, GeoJSON, PostgreSQL, vizualizácia geografických dát, dátový model, Geovisto, autentifikácia a autorizácia užívateľov, správa dát

Keywords

geographic data, geographic information system, Leaflet, web frameworks, user interface frameworks, user Interface, GeoJSON, PostreSQL, visualization of geographic data, data model, Geovisto, user authentication and authorization, data management

Citácia

FEŇKO, Šimon. *Informačný systém pre tvorbu geografických objektov*. Brno, 2023. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jiří Hynek, Ph.D.

Informačný systém pre tvorbu geografických objektov

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Jiří Hynek, Ph.D. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Šimon Feňko
30. júla 2023

Poďakovanie

Chcel by som touto cestou poďakovať pánovi Ing. Jiří Hynek, Ph.D., za vedenie a pomoc pri mojej práci. Takisto by som chcel poďakovať Damiánovi, Jakubovi, Dávidovi a Jánovi za konzultácie.

Obsah

1 Úvod **4**

2 Geografické dáta a práca s nimi **6**
2.1 Dáta . . . 6
2.2 Geografický informačný systém (GIS) . . . 7
2.2.1 Využitie GIS . . . 7
2.3 Práca s geografickými dátami . . . 7
2.3.1 Zobrazenie vektorových geografických dát . . . 9
2.3.2 Zobrazenie rastrových geografických dát . . . 11
2.3.3 Využitie geografických dát . . . 12
2.3.4 Spracovanie geografických dát . . . 12

3 Vizualizácia geografických dát **16**
3.1 Programové knižnice . . . 17
3.1.1 Leaflet . . . 17
3.1.2 MapBox . . . 18
3.1.3 OpenLayers . . . 19
3.2 Webové frameworky pro tvorbu UI . . . 20
3.2.1 Porovnanie UI frameworkov na základe popularity . . . 20

4 Analýza **23**
4.1 Typy uživateľov . . . 23
4.2 Súčasný stav . . . 24
4.2.1 Mapové nástroje . . . 24
4.3 Definícia požiadavkov . . . 26

5 Návrh **27**
5.1 Architektúra . . . 27
5.2 Návrh pohľadov a akcií . . . 28
5.3 Dátový model . . . 31

6 Implementácia **32**
6.1 Backend . . . 32
6.2 Frontend . . . 33
6.2.1 Navigačný panel . . . 33
6.2.2 Tabuľky geografických objektov . . . 34
6.2.3 Ukladanie geografických objektov . . . 35
6.2.4 Zdieľanie geografických objektov . . . 36

6.2.5 Tabuľka uživateľov 36
6.2.6 Registrácia uživateľa 37
6.2.7 Prihlasovanie uživateľa 39
6.3 Databáza 40

7 Testovanie **42**
7.1 Testovanie systému užívateľmi 42
7.1.1 Testovací adept 1 42
7.1.2 Testovací adept 2 42
7.1.3 Testovací adept 3 43
7.1.4 Návrh ďalšieho rozšírenia 43

8 Záver **44**

Literatúra **45**

Zoznam obrázkov

2.1 Tri časti geopriestorových dát 9
2.2 Základný objekt vektorového zobrazenia dát - bod 10
2.3 Základný objekt vektorového zobrazenia dát - úsečka 10
2.4 Základný objekt vektorového zobrazenia dát - polygon 11
2.5 Príklad rastrového zobrazenia dát 11

3.1 Zobrazenie nezamestnanosti v Európe v roku 2017 16
3.2 Zobrazenie maximálnych teplôt v Európe v dňoch Júl 17-23, 2022 17
3.3 Príklad mapy zoprazujúcej mesto Londýn pomocou knižnice Leaflet 18
3.4 Príklad mapy zoprazujúcej mesto Kodaň pomocou knižnice Mapbox 18
3.5 Príklad mapy zoprazujúcej štát Švajčiarsko pomocou knižnice OpenLayers . 19
3.6 Graf popularity medzi UI frameworkmi 21
3.7 Tabuľka základných rozdielov medzi React a Angular 22

4.1 Príklad finálneho widgetu mapy. Obsahuje bočný panel (vľavo), ktorý sa používa na konfiguráciu vrstiev, definíciu pravidiel filtrovania a všeobecné nastavenie mapy. Táto mapa obsahuje tri vrstvy: choroplet , marker a connection layer. Príklad ukazuje konfiguráciu vrstvy choropletu. Prepája dátovú doménu „od“ s vizuálnou dimenziou „krajina“, dátovú doménu „hodnota“ s vizuálnou dimenziou „hodnota“ a na agregáciu hodnôt používa funkciu „súčet“. 25

5.1 Domovská stránka systému pre neprihláseného uživateľa 28
5.2 Domovská stránka prihláseného uživateľa 29
5.3 Zobrazený GeoJSON pomocou tlačítka Get 29
5.4 Tabuľka uživateľov u prihláseného správcu 30
5.5 Tabuľka súborov daného uživateľa u správcu 30
5.6 Relačný model navrhovaného systému 31

6.1 Navigačný panel pre neprihláseného uživateľa 34
6.2 Navigačný panel pre prihláseného uživateľa 34
6.3 Tabuľka súborov typu GeoJSON daného uživateľa 35
6.4 Popup pre uloženie geojsonu 36
6.5 Popup pre zdieľanie geojsonu s iným uživateľom 36
6.6 Tabuľka uživateľov 37
6.7 Popup pre vytvorenie uživateľa 38
6.8 Registračný formulár 39
6.9 Prihlasovací formulár 40

Kapitola 1

Úvod

V dnešnej digitálnej ére predstavuje správa geografických dát a ich efektívne využitie kľúčovú oblasť pre mnoho odvetví, ako je napríklad geografia, urbanizmus, environmentálna analýza a mnohé ďalšie. Geografické informácie a ich vizualizácia zohrávajú dôležitú úlohu pri pochopení priestorovej distribúcie a interakcií medzi rôznymi geografickými objektmi.

Cieľom tejto diplomovej práce je navrhnúť a implementovať komplexný a funkčný informačný systém pre správu geografických objektov, ktoré budú vytvorené pomocou programovej knižnice Geovisto. Tento systém ktorý umožní užívateľom efektívne pracovať s geografickými dátami a zlepšiť tak ich schopnosť analyzovať a vizualizovať priestorové informácie má za úlohu umožniť užívateľom ukladať a zdieľať ich vlastné vytvorené geoJSONy, ale tiež importovať vlastné dáta, a to v prostredí, ktoré bude intuitívne a používateľsky prívetivé. Vďaka tejto práci bude možné efektívne pracovať s geografickými dátami a využívať ich v rôznych aplikáciách a projektových úlohách. Navrhovaný systém predstavuje cenný nástroj pre rôzne odvetvia, ktoré využívajú geografické dáta vo svojej každodennej práci a výskume.

V prvej časti práce je venovaná pozornosť základným pojmom týkajúcim sa geografických dát a ich spracovania. Ďalej sa zameriava na predstavenie geografického informačného systému, jeho využitia a rôznych spôsobov práce s geografickými dátami, ako je zobrazenie vektorových a rastrových geografických dát, ako aj ich spracovanie a analýza.

Druhá časť práce sa zameriava na vizualizáciu geografických dát, kde sú prezentované programové knižnice, ktoré slúžia na tvorbu interaktívnych mapových aplikácií. Medzi tieto knižnice patria Leaflet, MapBox a OpenLayers. Okrem toho sa v tejto časti práce zhodnocujú rôzne webové frameworky používané pre tvorbu užívateľských rozhraní, pričom je na nich založený aj výsledný informačný systém.

Analýza je základom každého návrhu, preto sa tretia časť práce venuje analýze potrieb a požiadaviek užívateľov, vytváraniu typov užívateľov a súčasnému stavu existujúcich mapových nástrojov. Na základe tejto analýzy sú následne definované požiadavky, ktoré musí informačný systém spĺňať.

V návrhovej časti práce je podrobne rozpracovaná architektúra systému, ktorá umožňuje efektívne a prehľadné usporiadanie jednotlivých komponentov. Ďalej sú navrhnuté pohľady a akcie, ktoré budú umožňovať užívateľom jednoduché ovládanie a manipuláciu s geografickými dátami. Okrem toho je vytvorený aj dátový model, ktorý definuje štruktúru dát a vzťahy medzi nimi.

Implementácia je kľúčovou časťou tejto práce, kde sú detailne opísané postupy a technológie využité pri tvorbe backendu a frontendu informačného systému. V tejto časti sú zahrnuté aj konkrétne kroky pri implementácii navigačného panelu, tabuľky súborov typu

GeoJSON, ukladania a zdieľania dát, ako aj mechanizmy registrácie a prihlásenia užívateľov. Systém je podporený robustnou databázou, ktorá je implementovaná za pomoci objektovo-relačného databázového systému PostgreSQL.

Kapitola 2

Geografické dáta a práca s nimi

V dnešnom modernom svete je veľká časť digitálnych dát tvorená geografickými dátami. A práve definícii geografických dát sa bude venovať prvá časť tejto kapitoly. Takisto sa bude rozoberať možnosť referencií na geografické objekty a popisovať možnosti, v akých formátoch je možné ich spracovať na to, aby ich bolo možné uživateľovi čitateľným spôsobom prezentovať.

2.1 Dáta

Na to, aby bolo jednoduchšie pochopiť, čo sú geografické dáta, je potrebné si najprv vysvetliť, čo sú to dáta a geografický informačný systém, ktorý sa označuje v skrátenej forme ako GIS. Podľa [5] sú dáta súbor diskrétnych hodnôt, ktoré vyjadrujú informácie, množstvo, kvalitu, skutočnosť, štatistiku alebo iné základné významové jednotky alebo jednoducho postupnosť symbolov, ktoré možno ďalej interpretovať.

Dáta sú zvyčajne usporiadané do rôznych štruktúr. Napríklad tabuľky, ktoré sa môžu samé používať ako dáta vo vyšších a väčších štruktúrach.

Podľa [12] dáta rozlišujeme na:

- **Nominálne** – sú také dáta, ktoré majú význam len určitej a istej kvality. Preto o nich často hovoríme ako o kvalitatívnych dátach. Jednotlivé hodnoty sa nedajú porovnávať. Keďže sú neporovnateľné, to znamená, že ich nedokáže zoraďovať a neexistuje pri nich nič také ako ich veľkosť. Medzi typické príklady takéhoto typu dát patrí napríklad krvná skupina alebo sekvencia DNA, to znamená, že sa jedná o druh kategórie. Z takého dôvodu u nich aj často hovoríme ako o kategoriálnych dátach.

- **Ordinálne** – sú také dáta, ktoré predstavujú podobne ako nominálne dáta výber z nejakého počtu možností. Významný rozdiel medzi ordinálnym a nominálnym typom dát je ten, že pri ordinálnom type sme schopní prirodzene vykonať usporiadanie. U každej dvojice hodnôt dokážeme ľahko určiť, ktorá hodnota je vyššia a ktorá nižšia. Typickým príkladom takéhoto typu dát je napríklad najvyššie dosiahnuté vzdelanie.

- **Intervalové** – sú také dáta, pri ktorých má význam hodnotiť aj vzdialenosť medzi jednotlivými kategóriami alebo hodnotami, na rozdiel od ordinálneho typu dát. Ideálnym príkladom takéhoto typu dát je napríklad teplota, ktorá je meraná v Celziovej teplotnej škále. Rozdiel teploty 10C je vždy rovnaký bez ohľadu na konkrétnu poťiatočnú teplotu. Nezahrňujú však pomery hodnôt. To znamená, že tvrdenie, ktoré

hovorí, že nárastom teploty z $10°C$ na $30°C$ sa teplota zvýšila o 3x, je z fyzikálneho hľadiska nesprávne a chybné.

- **Pomerové** – sú také dáta, u ktorých už sú zadefinované aj pomery jednotlivých hodnôt. Patria tu všetky fyzikálne veličiny, ktoré sú definované v súlade SI. Ich ďalšou charakteristickou vlastnosťou je, že majú jasne definovanú absolútnu nulu. Napríklad termodynamická absolutní nula, nulová vzdálenost alebo nulová hmotnost.

Pomerové a intervalové dáta bývajú zvyčajne spojité, ich hodnoty sa môžu plynule meniť v určitom intervale. Naopak dáta ordinálne a hlavne nominálne sú obvykle diskrétne. Dosahujú len len určitého konečného počtu možných hodnôt.

2.2 Geografický informačný systém (GIS)

GIS je podľa [2] nástroj, ktorý je využívaný na získavanie, analyzovanie, vizualizáciu a manažment dát s priestorovým alebo mapovým vyjadrením, ktoré sú základom takéhoto systému. Takisto obsahujú informácie o geografickej polohe a informácie, ktoré popisujú jednotlivé objekty.

Súčaťou GISu sú:

- Hardvér – počítače, servery alebo zariadenia na zber dát.
- Softvér – špecializované programy pre prácu s priestorovými dátami.
- Dáta – priestorové údaje obsahujúce informáciu o polohe a súvisiace popisné informácie, tvoria databázu priestorových informácií.
- Užívatelia – spracovatelia dát, administrátori GIS a prijímatelia priestorových informácií.

2.2.1 Využitie GIS

GIS sa využíva v rôznych oblastiach ľudskej činnosti, dá sa povedať, že sa využíva všade, kde je potrebné a nutné pracovať s priestorovou informáciou. Medzi najznámejšie príklady využitia GIS podľa [1] patria:

- digitálne mapovanie,
- správa zariadení,
- trhová a demografická analýza,
- doprava a logistika,
- dizajn a inžinierstvo.

2.3 Práca s geografickými dátami

Podľa [11] sú geografické dáta, tiež označované ako geodata alebo geopriestorové dáta, informácie o geografických lokalitách, ktoré možno uložiť a využívať v geografickom informačnom systéme (GIS). Vzťahujú sa na údaje a informácie, ktoré sú explicitne alebo implicitne spojené s polohou vzhľadom na Zem. Geografické dáta môžu pochádzať z telematických

zariadení, údajov Global Positioning System (GPS), geopriestorových satelitných snímok, Internet of Things a geotagging. Niektoré geografické témy, do ktorých možno geografické dáta zoskupiť, zahŕňajú:

Kultúrne témy

- **Administratívne dáta** – hranice (mestá a plánovanie).
- **Sociálno-ekonomické údaje** – demografia (hospodárstvo a kriminalita).
- **Doprava** – cesty (železnice a letiská).

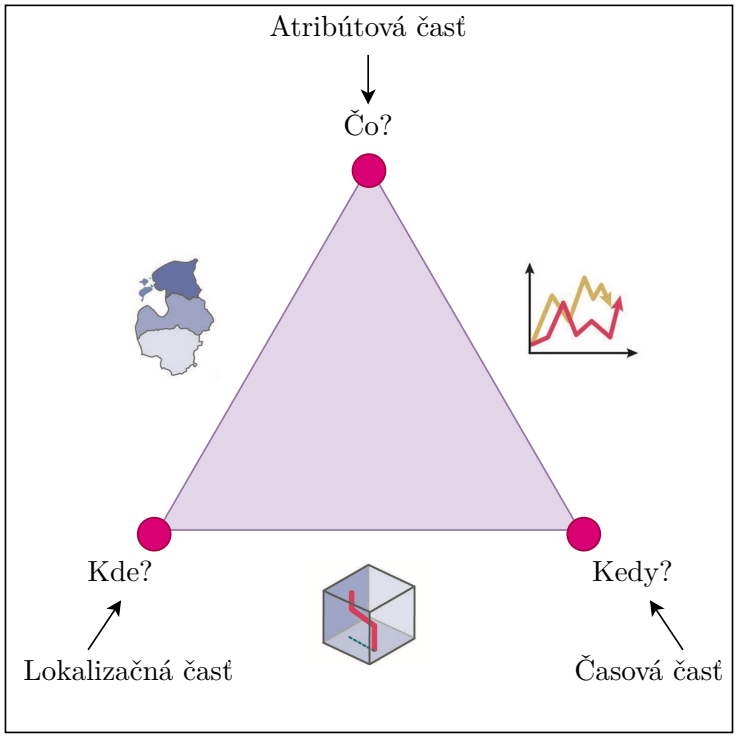
Fyzické témy

- **Údaje o nadmorskej výške** – terén a reliéf.
- **Environmentálne údaje** – poľnohospodárstvo - pôda a podnebie.
- **Hydrografické údaje** – oceány, jazerá a rieky.

Geografické dáta sa líšia od iných druhov dát v tom, že sa vzťahujú k určitému miestu alebo objektu na planéte a majú nejaký odkaz v priestore. Geografická poloha môže byť reprezentovaná priamo popisom geografickej za pomoci súradníc zemepisnej dĺžky a šírky, alebo na druhú stranu ako referencia na nejaký geografický objekt. Vďaka tomu dokážeme geodata zobraziť priamo na mapu. Geografické dáta sa teda odkazujú na geografické objekty za pomoci identifikátorov. Tie obsahujú jednotlivé geografické objekty. Následne geografické objekty môžu byť reprezentované v troch variantach ako body, čiary alebo polygóny. Podľa [11] sú geodata zvyčajne delené do troch častí:

- Lokalizačné údaje sa týkajú geometrických aspektov ako napríklad poloha a rozmery javu, o ktorom máme nejakú geometrickú informáciu.
- Časové údaje sa týkajú časového okamihu, pre ktorý sú platné údaje o polohe aj údaje o atribútoch.
- Atributové údaje sa týkajú iných negeometrických charakteristík.

Pomocou vyššie spomenutých troch častí (lokalizačné údaje, časové údaje a atributové údaje) dokážeme odpovedať na tri základné otázky ohľadom geopriestorových dát, ktoré stanovili, viď. obrázok č.*2.1*.



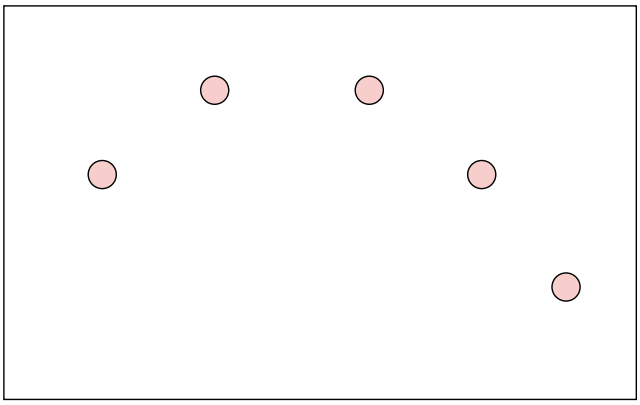
Obr. 2.1: Tri časti geopriestorových dát
[11]

V prípade rozhodovania ako zobraziť model dát reálneho sveta pomocou geografického informačného systému, sa musíme rozhodnúť, akú charakteristiku majú dané geografické objekty, ktoré se snažíme zobraziť. Existujú dva typy geografický objektov, ktoré sa snažíme zobrazovať. Jeden z dvoch typov sú vektorový dáta a druhý typ sú rastrové dáta.

2.3.1 Zobrazenie vektorových geografických dát

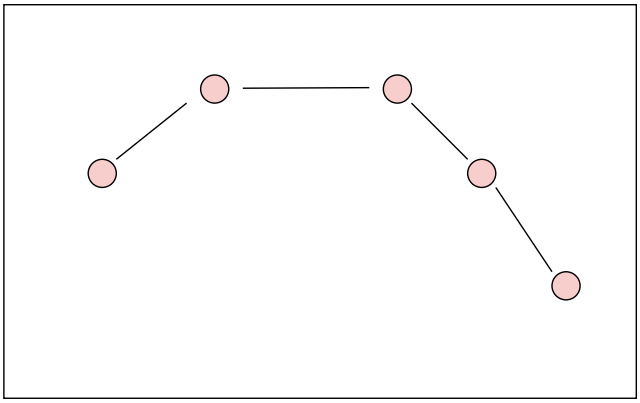
Podľa [8], sa vektorová grafika skladá z vrcholov a ciest. To znamená, že vektorové údaje nie sú tvorené mriežkou pixelov. Pri zobrazení vektorových dát využívame tri základné typy objektov: body, úsečky a mnohoúhelníky (polygóny).

- Body – vektorové body sú súradnice XY. V skratke to znamená, že sú to zemepisná šírka a dĺžka s priestorovým referenčným rámcom. V prípade ak sú prvky príliš malé na to, aby sa dali reprezentovať ako polygóny, používajú sa body. Napríklad v globálnej mierke nie je možné zobraziť hranice mesta. V tomto prípade sa na zobrazenie miest na mapách často používajú body, viď. oprázok č. *2.2*.



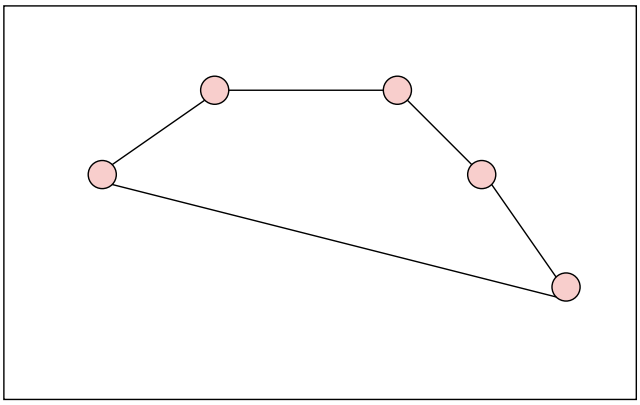
Obr. 2.2: Základný objekt vektorového zobrazenia dát - bod

- Úsečky – vektorové úsečky spájajú jednotlivé vrcholy s cestami. V podstate spájajú body v stanovenom poradí a vzniká vektorová úsečka, pričom každý bod predstavuje vrchol, viď. oprázok č. *2.3*. Línie zvyčajne predstavujú prvky, ktoré sú spojené. Napríklad na mapách sa ako vektorové čiary zobrazujú rieky, cesty a potrubia. Rušnejšie diaľnice majú často hrubšie čiary ako opustené cesty.



Obr. 2.3: Základný objekt vektorového zobrazenia dát - úsečka

- Polygón – keď sa spojí sada vrcholov v určitom poradí a uzavrú sa, ide v takom prípade o vektorový polygónový prvok. Pri vytváraní polygónu sú prvý a posledný pár súradníc rovnaké, viď. oprázok č. *2.4*. Kartografi používajú polygóny na zobrazenie hraníc a všetky majú plochu. Napríklad pôdorys budovy má štvorcovú plochu a poľnohospodárske polia majú výmeru.



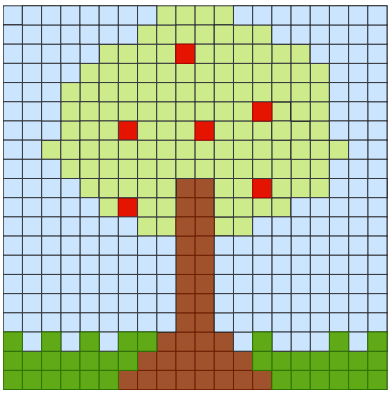
Obr. 2.4: Základný objekt vektorového zobrazenia dát - polygon

Výhodou vektorovej reprezentácie je, že vektorové dáta majú vrcholy a cesty, znamená to, že grafický výstup je vo všeobecnosti estetickejší. Okrem toho poskytuje vyššiu geografickú presnosť, pretože údaje nie sú závislé od veľkosti mriežky.

Naopak nevýhodou je, že spojité údaje sa zle ukladajú a zobrazujú ako vektory. Ak chceme zobraziť spojité údaje ako vektor, vyžadovalo by si to značnú generalizáciu. Hoci je topológia pre vektorové údaje užitočná, je často náročná na spracovanie. Akékoľvek úpravy prvkov si vyžadujú aktualizácie topológie. Pri veľkom množstve funkcií sú algoritmy na manipuláciu s vektormi zložité.

2.3.2 Zobrazenie rastrových geografických dát

Podľa [8] sa rastrové dáta skladajú z pixelov (označovaných aj ako bunky mriežky) viď. oprázok č. *2.5*. Zvyčajne sú pravidelne rozmiestnené a štvorcové, ale nemusia byť. Rastre často vyzerajú pixelovo, pretože každý pixel má svoju vlastnú hodnotu alebo triedu.



Obr. 2.5: Príklad rastrového zobrazenia dát

Výhodou rastrovej reprezentácie je vykonávanie rôznych analýz a ťažby dát. Rastrové modely sú užitočné na ukladanie údajov, ktoré sa priebežne menia. Napríklad výškové plochy, teplota a kontaminácia olovom.

Nevýhodou môže napríklad byť to, že súbory rastrových údajov môžu byť potenciálne veľmi veľké, pretože zaznamenávajú hodnoty pre každú bunku v obraze. So zvyšujúcim sa rozlíšením sa veľkosť bunky zmenšuje. To je však na úkor rýchlosti spracovania a ukladania údajov.

2.3.3 Využitie geografických dát

Geografické údaje sa podľa [11] používajú na vizuálne zobrazenie a lepšie pochopenie vplyvu ľudskej činnosti na základe konkrétnej geografickej polohy. Geografické informačné systémy používajú digitálny softvér na zhromažďovanie, ukladanie a analýzu geografických údajov, ktoré sa potom používajú na vytváranie máp vrstiev na lepšiu analýzu komplexných environmentálnych udalostí a sociálno-ekonomických trendov.

Vizuálna prezentácia údajov s geografickým kontextom pomáha objasniť, ako údaje súvisia s konkrétnym miestom, a osvetľuje vzory, ktoré by inak mohli zostať nezistené. Vizualizácia geodát sa dosahuje pomocou geopriestorového modelovania, ktoré využíva pokročilé kartografické technológie na integráciu interaktívnej vizualizácie do tradičných máp a poskytuje analytikom možnosť interakcie, zmeny parametrov a identifikácie vzťahov na mape geodát.

Aplikácia geografických údajov v geopriestorových technológiách je užitočná najmä v prípadoch, ako je plánovanie a rozvoj miest, určovanie trasy pre letecké spoločnosti, hodnotenie majetkových rizík pre poisťovníctvo, výstrahy pri evakuácii v súvislosti s počasím, optimalizácia vojenskej logistiky, rýchla identifikácia a oprava sieťových anomálií a telekomunikácie.

2.3.4 Spracovanie geografických dát

Geografické dáta môžu byť spracovávané v rôznych formátoch. Následujúca sekcia sa bude venovať daným formátom.

GeoJSON

GeoJSON [3] je open standard formát určený na reprezentáciu jednoduchých geografických prvkov spolu s ich nepriestorovými atribútmi. Je založený na formáte JSON. Je plne kompatibilný s už spomenutým formátom JSON. Vďaka tomu, je ľahko použiteľný v prostredí webových prehliadačov. Takisto je aj prehľadný pre bežného uživateľa. GeoJSON špecifikuje niekoľko geometrických typov:

- **Point** – reprezentácia bodu na mape za pomoci súradníc, slúži na presnú lokalizáciu. Poloha sa udáva za pomoci zemepisnej šírky a dlžky (adresy a miesta).
- **LineString** – ide o krivku, ktorá je reprezentovaná zoradeným poľom súradníc počiatočného a koncového bodu (ulice, diaľnice a hranice).
- **Polygon** – mnohouholník, obsahuje súradnice jednotlivých vrcholov mnohouhelníkov o minimálne štyroch bodoch, ktoré sú reprezentované pomocou typu LineString. Platí, že prvá a posledná súradnica je identická (krajiny, provincie, pozemky a iné).

- **Multipoint** – kolekcia bodov na mape, ktoré majú rovnaké vlastnosti a tie sú zahrnuté do jedného objektu.
- **MultiLineString** – kolekcia viac objektov kriviek, ktoré majú rovnaké vlastnosti. Obsahuje pole súradníc vrcholov jednotlivých mnohouholníkov.
- **MultiPolygons** – kolekcia viacerých polygónov, ktoré majú rovnaké vlastnosti.
- **GeometryCollections** – kolekcia geometrických objektov, ktorá obsahuje kolekciu ľubovolný počet ľubovolných vyššie špecifikovaných typov a tak ich združuje do väčších objektov.

Na príklade č. *2.1* je možné vidieť ukážku objektu `Feature` obsahujúci geometrický objekt bod.

```
{
    "type": "Feature",
    "geometry": {
        "type": "Point",
        "coordinates": [125.6, 10.1]
    },
    "properties": {
        "name": "Dinagat Islands"
    }
}
```

Výpis 2.1: Príklad kódovania dát vo formáte GeoJSON.

Výhody používania formátu GeoJSON sú, že GeoJSON je otvorený štandard, čo znamená, že je voľne dostupný a môže ho používať ktokoľvek. Vďaka tomu je prístupný širokému okruhu používateľov a vývojárov. GeoJSON je dátový formát, ktorý je efektívny na ukladanie a prenos veľkého množstva geopriestorových údajov a takisto používa ľudsky čitateľný formát JSON, ktorý uľahčuje prácu s ním a jeho pochopenie.

Nevýhodou môže byť pri väčších veľkostiach súborov, pretože v porovnaní s binárnymi formátmi môžu mať súbory GeoJSON väčšiu veľkosť, čo môže byť náročné na prácu s veľkými súbormi údajov.

ShapeFile

Podľa [7] je shapefile formát geopriestorových vektorových údajov pre softvér GISu. Formát shapefile môže priestorovo popisovať vektorové prvky: body, čiary a polygóny, ktoré predstavujú napríklad vodné studne, rieky a jazerá. Každý prvok má zvyčajne atribúty, ktoré ho opisujú, napríklad názov alebo teplotu. Vďaka tomu, že neobsahuje topologické dáta ako ostatné formáty, tak je formát Shapefile rýchlejší vo vykresľovaní a v možnosti editácie oproti ostatným formátom.

Výhodou používania formátu Shapefile môže byť jeho široká podpora, pretože je široko používaný a dobre podporovaný formát geopriestorových údajov, čo znamená, že sa dá ľahko používať a zdieľať medzi rôznymi aplikáciami a systémami GIS a takisto výhodou je jeho efektívne ukladanie. Shapefile je navrhnutý ako efektívny formát na ukladanie veľkého množstva geopriestorových údajov, takže je vhodný na prácu s veľkými súbormi údajov.

Nevýhodou môže byť obmedzená funkčnosť, pretože v porovnaní s inými formátmi geopriestorových údajov, ako je GeoJSON, je funkčnosť Shapefile obmedzená. Nemusí byť vhodný pre komplexné aplikácie GIS, ktoré vyžadujú pokročilé funkcie.

KML

KML (*Keyhole Markup Language*) je podľa [16] jazyk založený na XML, ktorý sa používa na zobrazovanie geopriestorových údajov v mapovom prehliadači, ako sú Google Earth, Google Maps a NASA WorldWind. Používa sa na špecifikáciu polohy, vzhľadu a správania dát v geografickom informačnom systéme. KML sa môže používať na reprezentáciu bodov, čiar, polygónových tvarov, obrázkov, 3D modelov a iných geopriestorových údajov. Je to otvorený štandard a môže sa používať v rôznych aplikáciách, napríklad v mapových a geografických informačných systémoch.

Výhodou je, že KML je otvorený štandard, čo znamená, že je voľne dostupný a môže ho používať ktokoľvek. Vďaka tomu je prístupný širokému okruhu používateľov a vývojárov. Takisto výhodou je Interoperabilita. To znamená, že je navrhnutý tak, aby bol kompatibilný s rôznymi aplikáciami a systémami, čo umožňuje zdieľanie údajov a spoluprácu na projektoch.

Naopak nevýhodou môže byť obmedzená funkčnosť, keďže KML je síce užitočný na zobrazovanie geopriestorových údajov, ale nemusí byť vhodný pre komplexné aplikácie GIS, ktoré vyžadujú pokročilé funkcie, obmedzená podpora, keďže KML je síce široko používaný, ale nemusia ho podporovať všetky aplikácie a systémy GIS a tiež môžu nastať problémy s výkonom, pretože veľké súbory KML sa môžu pomaly načítavať a zobrazovať, čo môže byť nevýhodou pri práci s veľkými súbormi údajov.[16]

Na ukážke č. *2.2* je možné vidieť príklad formátu KML, ktorý zobrazuje bod na mape. Takisto zobrazuje aj popis daného bodu, , ktorý následně môže geografický informačný systém pri danom bode zobrazovať.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">

    <Placemark>

        <name>Jednoduchy bod na mape</name>
        <description>Popis najdeneho bodu na mape</description>

        <Point>
            <coordinates>8.54,47.36,0</coordinates>
        </Point>

    </Placemark>

</kml>
```

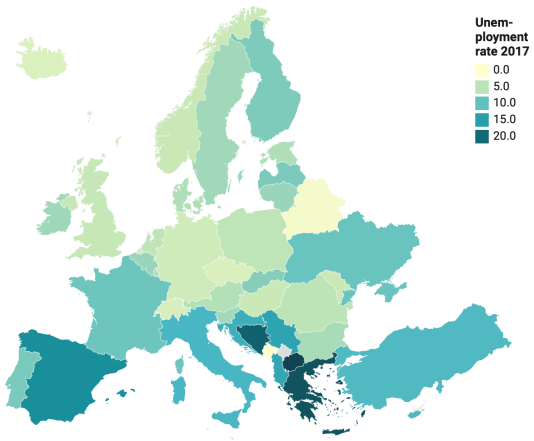
Výpis 2.2: Príklad formátu KML.

Kapitola 3

Vizualizácia geografických dát

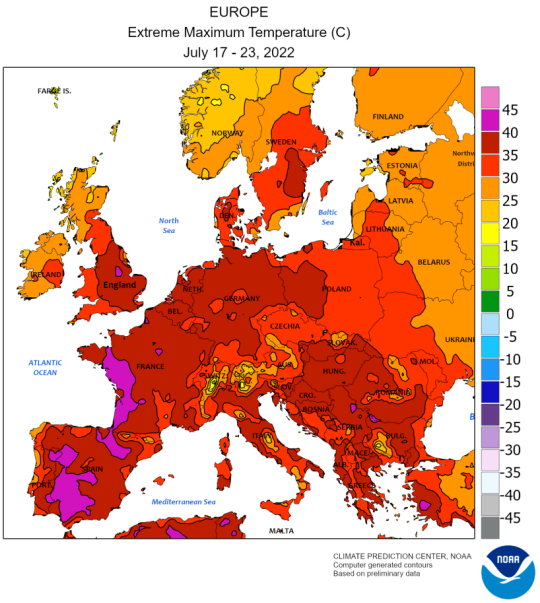
V dnešnej modernej dobe sa čím ďalej tym menej využívajú geografické materiály, ktoré sú v školských atlasoch alebo v papierovej podobe. Ako doba napreduje, začali sa využívať stále viac a viac v elektronickej podobe (ako software), na internete alebo inej aplikácii. Webová geopriestorová vizualizácia sa používa na zobrazenie geografických údajov na mape, čo používateľom umožňuje intuitívnejšie a zmysluplnejšie pracovať s údajmi a skúmať ich. Vizualizáciou údajov na mape môžu používatelia vidieť vzory, trendy a vzťahy, ktoré by bolo ťažké rozoznať len zo surových údajov. Webová geopriestorová vizualizácia je užitočná v širokej škále oblastí, ako sú okrem iného mestské plánovanie, environmentálna analýza, doprava a nehnuteľnosti.

Jedným zo spôsobov, ako je možné informácie predať, je reprezentácia v podobe tématických máp. Podľa [6] sa tématické mapy zameriavajú na ilustráciu typicky jednej charakteristiky dát a vkladá ich do kontextu geografického objektu. Využívajú k tomu grafické prvky (body, mnohouholníky, apod.) nad určitým mapovým podkladom. Podľa [6] je kartogram (viď obrázok č. *3.1*) alebo choropletová mapa je typ tematické mapy, ktorý je jednou z najčastejšie používaných máp v geopriestorových dátach. Používa farby alebo iné vizuálne prvky na zobrazenie štatistických dát o určitej oblasti na mape. V choropletovej mape sa jednotlivé oblasti zvyčajne farbia rôznymi farbami alebo odtieňmi farieb, ktoré zobrazujú rôzne úrovne hodnôt daných dát v týchto oblastiach.



Obr. 3.1: Zobrazenie nezamestnanosti v Európe v roku 2017

Ďalšou variantou tématickej mapy je teplotná mapa (anglicky *Heat map*) (viď obrázok č. *3.2*). Teplotná mapa reprezentuje intenzitu výskytu udalosti v rámci dátovej sady. Podobne ako kartogram, tak aj teplotná mapa využíva variácie zafarbenia k vyjadreniu intenzity, narozdiel od kartogramov, ale rieši iba frekvenciu výskytu v určitom konkrétnom bode súradnicového systému.



Obr. 3.2: Zobrazenie maximálnych teplôt v Európe v dňoch Júl 17-23, 2022

Webová geopriestorová vizualizácia často obsahuje interaktívne funkcie, ktoré umožňujú používateľom skúmať údaje a manipulovať s nimi v reálnom čase, čo môže zlepšiť pochopenie a rozhodovanie. Výhodou je prístupnosť, pretože sprístupňuje údaje širšiemu publiku, pretože sa dajú ľahko zdieľať a prezerať online.

3.1 Programové knižnice

K dispozícii na internete existuje obrovské množstvo knižníc, ktoré poskytujú základné rozhranie pre vytvorenie webových geografických vizualizací a ich následné vloženie do webových stránok. Ďalšia časť sa bude venovať skúmaniu a porovnávaniu geovizualizačných knižníc, ktoré sú v dnešnej dobe často používané.

3.1.1 Leaflet

Mapovacia knižnica Leaflet je mapovacia knižnica JavaScript s otvoreným zdrojovým kódom, ktorá sa používa na vytváranie interaktívnych máp vhodných pre mobilné zariadenia, viď obrázok č. *3.3*. Podľa [13] má veľkosť len 38 KB a obsahuje väčšinu mapovacích funkcií, ktoré vývojári potrebujú na použitie vo svojich projektoch mobilných mapovacích aplikácií. Funguje takmer na všetkých existujúcich desktopových a mobilných platformách a je škálovateľná pomocou zásuvných modulov. Používa sa aj na vykresľovanie vektorových a rastrových máp, napríklad OpenStreetMaps, respektíve MapBox.

Mapa vytvorená pomocou Leaflet.js sa skladá z niekoľkých vrstiev, s ktorými môže vývojar pracovať a nanášať na ne vlastné geografické objekty. Môžu to byť napríklad polygony

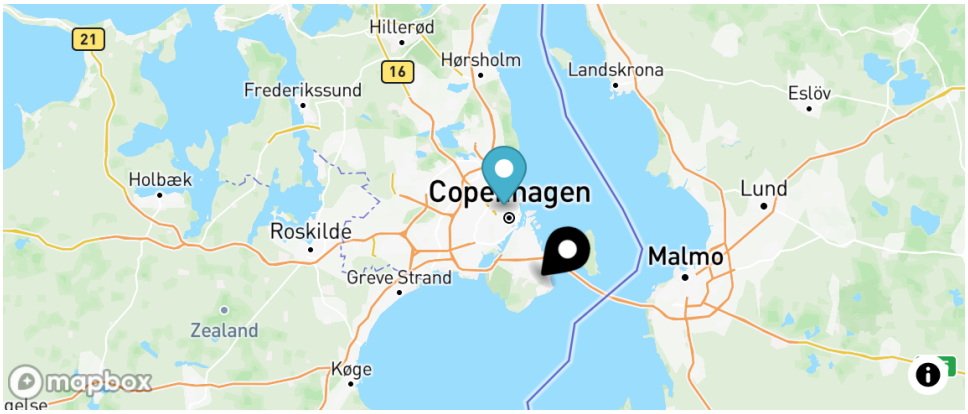
alebo body, ktoré sú prevažne v podporovanom formáte GeoJSON. Definície objektov je možné programovo meniť. Vďaka tomu sme schopný meniť ich atribúty (rozmer, farba, poloha).

Obr. 3.3: Príklad mapy zoprazujúcej mesto Londýn pomocou knižnice Leaflet

Medzi hlavné výhody Leaflet.js [13] patrí jeho uživateľská prívetivosť. Je navrhnutý tak, aby sa ľahko používal a bol intuitívny, takže je prístupný širokému okruhu používateľov vrátane tých s obmedzenými technickými znalosťami. Je navrhnutý tak, aby bol ľahký a efektívny, vďaka čomu je ideálny na používanie v mobilných zariadeniach a vo webových aplikáciách s obmedzenými zdrojmi. Výhodou je aj najmä široké uplatnenie, pretože je široko rozšírený a má veľkú a aktívnu komunitu používateľov, vďaka čomu je ľahké nájsť podporu a zdroje pre vývoj s touto knižnicou. Jednotlivé objekty možno pridávať do skupín, ktoré si vývojár definuje. Takto zoskupené objekty možno upravovať naraz. K dispozícii je aj možnosť pridania vlastných ovládacích prvkov, ako sú tlačidlá alebo napríklad prvky s funkciou skrytia alebo zobrazenia objektov alebo vrstiev.

3.1.2 MapBox

Podľa [4] je Mapbox online platforma, ktorá umožňuje jednoduché zobrazenie týchto máp na viacerých platformách, pretože nevyžaduje stiahnutie žiadneho ďalšieho softvéru. Všetky potrebné knižnice a nástroje potrebné na vykresľovanie mapy sú dostupné v rámci Mapboxu, viď obrázok č. *3.4*.



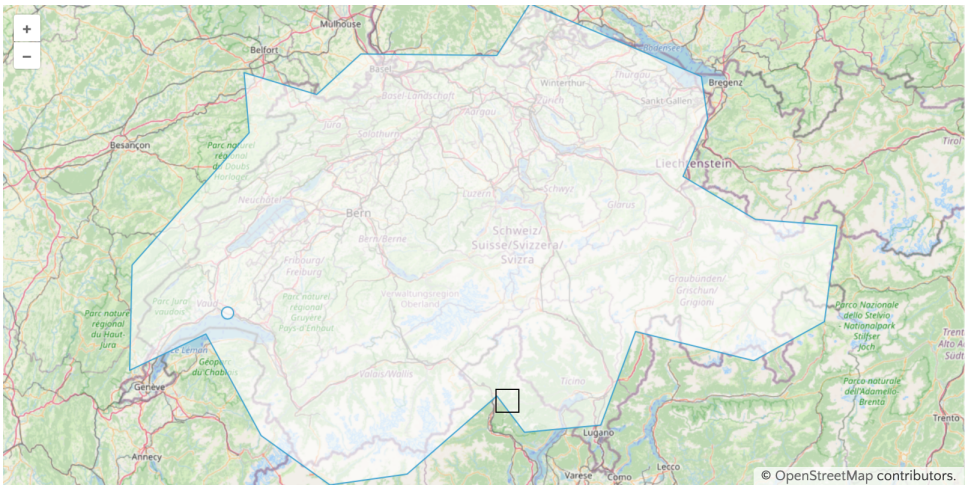
Obr. 3.4: Príklad mapy zoprazujúcej mesto Kodaň pomocou knižnice Mapbox

Je open source, čo je veľkou výhodou, pretože na jeho stránkach je veľa príkladov, ktoré môžu pomôcť pri vytváraní nových máp. Majú tiež mnoho návodov, ako krok za krokom pracovať s údajmi a ako používať všetky ich rôzne nástroje na získanie čo najlepších máp.

Naopak nevýhodou tejto platformy sú náklady, ktoré sú potrebné pre prácu s ňou. Mapbox môže byť drahší ako iné mapové riešenia, najmä pri veľkom objeme použitia alebo pokročilých funkciách. Systém môže byť príliš komplexný pre jednoduchšie projekty. Príklady z dokumentácie bývajú niekedy nedostatočné na pochopenie.

3.1.3 OpenLayers

OpenLayers je modulárna a výkonná knižnica, pomocou ktorej dokážeme vytvárať interaktívne webové mapy, ktoré sa dajú zobraziť takmer v každom webovom prehliadači, viď obrázok č. *3.5*. Nevyžaduje akýkoľvek špecialny softvér ani nastavenia na strane servera. To znamená, že ju dokážeme používať bez toho, aby sme si museli čokoľvek stiahnuť. Pôvodne ju vyvinula spoločnosť Metacarta ako čiastočnú reakciu na Mapy Google, ale postupne sa z nej stal vyspelý a populárny rámec s mnohými nadšenými vývojármi a veľmi nápomocnou komunitou. Vďaka knižnici OpenLayers je možné vytvárať jednoduché mapy, ktoré zobrazujú značky, ale aj náročnejšie a zložitejšie mapy, kde sú zobrazované napríklad trasy letov a ich animácie. Objekty sa definujú programovou formou, kedy sa zadávajú objektu geografické súradnice a atribúty a tým sa na danom mieste zobrazí. Alebo je možné definovať objekty dynamickou formou, kedy sa sa objekty nanášajú na mapu pomocou rozhrania. Základnou funkciou je aj možnosť kreslenia a s tými objektmi je možné manipulovať (meniť rotáciu alebo veľkosť).



Obr. 3.5: Príklad mapy zoprazujúcej štát Švajčiarsko pomocou knižnice OpenLayers

Hlavnou výhodou je, že OpenLayers umožňuje jednoduché vytváranie výkonných webových mapových aplikácií. Je veľmi výkonný, ale zároveň sa ľahko používa. To znamená, že uživateľ nepotrebuje žiadne veľké skúsenosti s programovaním na to, aby vytvoril fungujúcu a plnohodnotnú mapu. Takisto je open-source, je zadarmo a stojí za ním silná komunita.

Jednou z nevýhod OpenLayers je, že v porovnaní s inými mapovacími knižnicami môže mať OpenLayers relatívne dlhý čas načítavania zložitých máp a veľkých súborov údajov, čo vedie k pomalšiemu výkonu na starších zariadeniach.

3.2 Webové frameworky pro tvorbu UI

Webové frameworky pre používateľské rozhranie sú softvérové rámce, ktoré sú určené na pomoc vývojárom pri vytváraní dynamických a interaktívnych používateľských rozhraní (UI) pre webové aplikácie. Tieto rámce poskytujú súbor nástrojov, knižníc a komponentov, ktoré zjednodušujú proces vytvárania používateľského rozhrania, čím vývojárom uľahčujú a urýchľujú vytváranie a správu komplexných webových aplikácií. Medzi obľúbené webové rámce pre používateľské rozhranie patria React, Angular, Vue.js, Ember.js a Svelte.

V súčasnosti pri implementácii front-end vrstvy webovej aplikácie vývojári často volia niektorý z JavaScriptových wébových frameworkov pre tvorbu UI, pretože umožňujú dekomponovať grafické uživateľské rozhranie do logických celkov (komponentov).

Webové frameworky v posledných rokoch neustále rastú na svojej popularite, protože v praxi so sebou prinášajú mnoho výhod. Medzi najhlavnejšie patria:

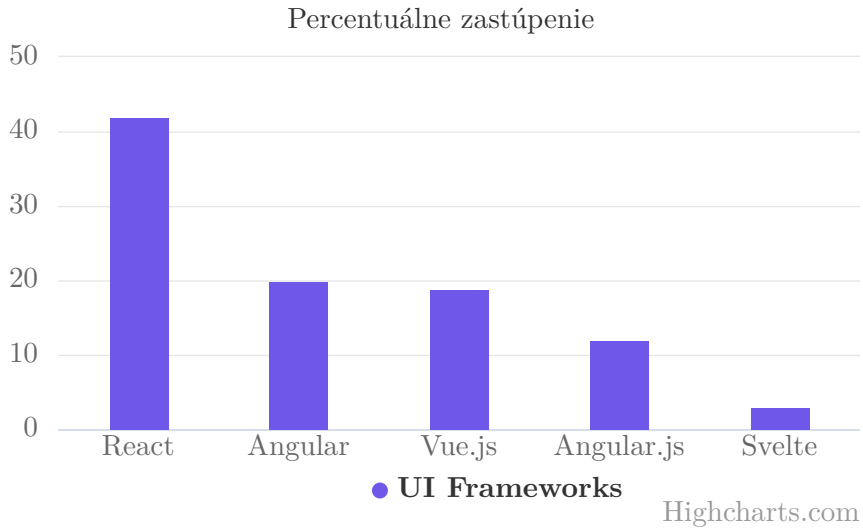
- **1. Produktivita** – webové rámce poskytujú sadu predpripravených komponentov používateľského rozhrania a nástrojov, ktoré vývojárom uľahčujú rýchle a efektívne vytváranie komplexných prvkov používateľského rozhrania.
- **2. Znovupoužiteľnosť** – vďaka webovému rámcu môžu vývojári opätovne používať kód a komponenty používateľského rozhrania, čo pomáha skrátiť čas a náklady na vývoj.
- **3. Kompatibilita s rôznymi prehliadačmi** – mnohé webové rámce majú zabudovanú kompatibilitu s rôznymi prehliadačmi, vďaka čomu vývojári nemusia písať vlastný kód pre každý prehliadač.
- **4. Spoľahlivý a udržateľný kód** – zdrojový kód, ktorý je implementovaný za použitia webových frameworkov dodržuje vhodné koncepty, ktorými sú organizácie zdrojového kódu, členenie projektu alebo aj jednoduchšia testovatelnosť.

Napriek mnohým výhodám, ktoré ponúkajú webové frameworky, sa s nimi tiahnu aj niektoré nevýhody. Môžu byť zložité a vývojárom môže trvať, kým sa v ich používaní zdokonalia. Takisto nevýhodou môže byť závislosť na aktualizáciách, pretože aplikácie vytvorené pomocou webových frameworkov sú závislé na aktualizáciách a bezpečnostných opravách daného frameworku. To môže spôsobiť problémy, ak sa rámec už aktívne neudržiava alebo ak aktualizácie porušia existujúcu funkcionalitu.

3.2.1 Porovnanie UI frameworkov na základe popularity

Server Stack Overflow[1] vykonal v máji v roku 2022 prieskum popularity medzi vývojármi [14], do ktorého sa zapojilo viac ako 70 000 respondetov po celom svete. Výsledkom bol aj výstup prieskumu najvyššej popularity medzi webovými frameworkmi. Na obrázku č. *3.6* je možné vidieť výsledok vo forme percentualného zastúpenia prvých zástupcov z rady JavaScriptových UI Frameworkov. Najpopulárnejší medzi vývojármi je framework React a to s 42,62%, ďalej následujú Angular s 20,39% hlasov, Vue.js s hodnotou 18,82% hlasov. Na následujucom mieste sa umiestnil framework, ktorý je predchodca Angularu a to Angular.js a to s hodnotou 11,49% hlasov. Na poslednom mieste sa umiestnil framework Svelte s 2,75% zastúpením. Z toho je možné odvodiť, že dva najpopularnejšie frameworky sú React a Angular.

[1] Viac informácií nájdete na: https://survey.stackoverflow.co/2022/#most-popular-technologies-webframe



Obr. 3.6: Graf popularity medzi UI frameworkmi

Ako bolo už vyššie spomenuté, vývojár má na výber pri rozhodovaní, aký UI framework využije širokú škálu. Napriek tomu, existuje zopár, ktoré sú dlhodobo najpoužívanejšie a z toho dôvodu ich možno považovať za najobľúbenejšie. Na základe prieskumu, ktorý bol vyššie spomenutý som sa rozhodoval a zvažoval prácu s frameworkom React a Angular, pretože boli prvé dva najpopulárnejšie. Hoci sú oba rámce veľmi účinné a široko používané, majú určité rozdiely, viď. obrázok č. *3.7* na základe ktorých som sa rozhodoval.

React využíva architektúru založenú na komponentoch, ktorá vývojárom umožňuje vytvárať komponenty používateľského rozhrania, ktoré možno ľahko opakovane používať a kombinovať na vytváranie komplexných používateľských rozhraní. Angular tiež používa architektúru založenú na komponentoch, ale má aj názorovo vyhranenejší a štruktúrovanejší prístup k architektúre aplikácie. React používa jednosmerný prístup viazania údajov, pri ktorom zmeny v modeli aktualizujú zobrazenie, ale zmeny v zobrazení neaktualizujú model. Angular používa obojsmerný prístup viazania údajov, kde sa zmeny v modeli a zobrazení automaticky premietajú do seba navzájom. Takisto výhodou je aj výkon, pretože sa React vo všeobecnosti považuje za systém s miernou výkonnostnou výhodou, pretože aktualizuje len tie časti používateľského rozhrania, ktoré sa zmenili, a nie znovu vykresľuje celé používateľské rozhranie. Vo väčšine aplikácií je však rozdiel vo výkone často zanedbateľný.

Parametre	React	Angular
Typ	Knižnica	Framework
Náročnosť	Jednoduchší ako Angular	Normálna
Architektúra	Redux/Flux	MVC
DOM	Virtual DOM	Real DOM
Abstrakcia	Silná	Stredná
Data-binding	Jednosmerné	Dvojsmerné
Autor	Facebook	Google
Jazyk	JavaScript + JSX	JavaScript + HTML

Obr. 3.7: Tabuľka základných rozdielov medzi React a Angular

Kapitola 4

Analýza

V tejto kapitole sa budem zameriavať na formuláciu cieľovej kategórie uživateľa a jeho požiadavkov. Pomocou nich bude definovaný problém a vykonané riešenie. Následne opíšem možných užívateľov výsledného informačného systému.

4.1 Typy uživateľov

- **Architekti a kartografi** – oba typy uživateľov budú využívať výslednú aplikáciu. Architekti hlavne na ohraničenie pôdorysov a využívať hlavne geometrické objekty, ktoré budú reprezentovať plány budov a rozličné miestosti. Kartografi zase pri vyznačovaní štátov alebo okresov a budú pracovať s objektami, ktoré budú reprezentovať územia.

- **Programátori** – Plánovanie telekomunikačných systémov je ďalšou oblasťou, v ktorej môžu byť geopriestorové údaje informatívne. Napríklad údaje o pešej premávke môžu poskytovateľom pomôcť pri plánovaní infraštruktúry okolo najrušnejších oblastí v susedstve. To im môže umožniť dosiahnuť čo najširšie pokrytie s čo najmenším množstvom hardvéru. Môže tiež informovať o cene, ktorú účtujú za zriadenie hotspotov WiFi, pričom inštalácie v oblastiach s vyššou návštevnosťou sú drahšie.

- **Verejný sektor** – Techniky a aplikácie geopriestorových údajov môžu byť užitočné, ak ste vo vláde alebo vo verejnom sektore. Môžete robiť správne rozhodnutia v oblasti komunitného plánovania na základe údajov o tom, kde sú budovy (a čo sú), kde ľudia žijú, kam zvyčajne za deň idú a akými cestami sa tam dostanú. Môžete napríklad naplánovať výstavbu ciest alebo iných dopravných systémov, aby ste skrátili časy chôdze/jazdy na miesta, ktoré ľudia pravidelne navštevujú, ako sú obchody s potravinami. Alebo môžete robiť veci opačne a naplánovať výstavbu dôležitých zariadení, ako sú nemocnice alebo školy, v oblastiach, ktoré sú už prístupné a sú veľmi frekventované.

- **Študenti a výskumníci** – Knižnica môže byť užitočná pre študentov a výskumníkov, ktorí pracujú s geografickými dátami. Tento nástroj môže pomôcť študentom a výskumníkom vytvárať a upravovať súbory vo formáte GeoJSON bez nutnosti zvláštnych technických zručností.

4.2 Súčasný stav

Geovisto je open-source projekt a programová knižnica, ktorá bola navrhnutá a vytvorená na pôde Fakulty informačných technológií v Brně a Institutu výpočetní techniky Masarykovy univerzity v Brně [9]. Bola implementovaná za pomoci knižníc Leaflet a D3.js. Knižnica Geovisto kombinuje programový prístup s možnosťou úpravy pripravených šablón pomocou uživateľského rozhrania. Táto vlastnosť, patrí medzi jedne z najväčších výhod tejto knižnice, pretože uživateľovi je umožnené vytvárať geopriestorové vizualizácie bez toho, aby potreboval nejaké znalosti programovania. Je teda zameraná na jednoduché použitie pre prefosionálov ale aj úplných začiatočníkov.

Architektúru knižnice Geovisto môžeme rozdeliť na dve základné časti:

1. **Jadro**: je vstupným bodom a venuje sa spracovaniu vstupných dát, ktoré poskytuje uživateľ a následne sa venuje inicializácii globálneho stavu mapy. To znamená, že zabezpečuje manipuláciu s mapou za pomoci Leaflet API a spracuváva dátove sady s geografickými objektami a konfigurácie.

2. **Mapové nástroje**: cieľom mapových nástrojov je pridávať ďalšie funkcionality jadru ako sú napríklad ovládacie prvky, mapové vrstvy či možnosti pre dalšie zpracovanie dát (filtre). Vďaka využitia modulov v architektúre knižnice je umožnené jednoduché pridávanie ďalších funkcionalít v budúcnosti.

4.2.1 Mapové nástroje

Základnou časťou systému je mapa. Tá má na starosti geovizualizáciu, a teda sa stará o zobrazovanie geografických dát v priestore. Následne nižšie sú popísané nástroje viď obrázok č. *4.1*, ktoré možno využiť.[10]

- **Sidebar** – nástroj, ktorý slúži na konfiguráciu vrstiev, definovanie pravidiel filtrovania a všeobecné nastavenie mapy. Tento nástroj je možné schovať a poskytuje API pre pridanie nových položiek do panelu.

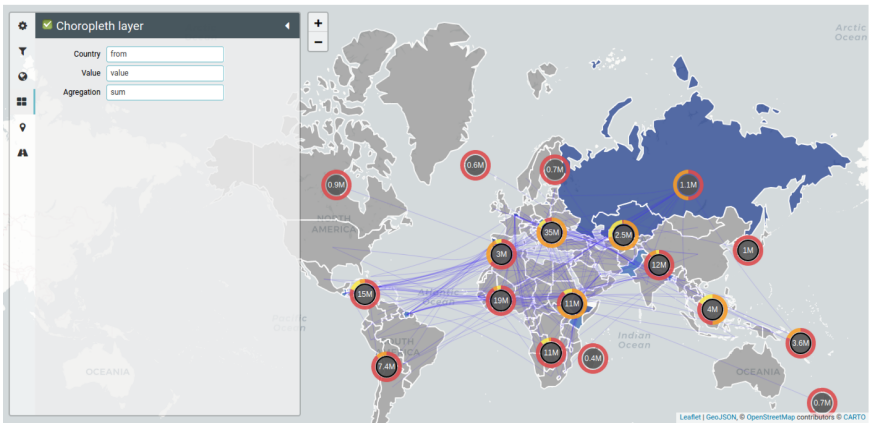
- **Choropleth Layer** – vrstva poskytuje možnosť použiť špecifikácie GeoJSON polygónov reprezentujúcich geograické regióny a prepojiť ich s údajmi. Primárne pracujeme so špecifikáciou svetových krajov. Môžu sa však použiť rôzne súbory GeoJSON, ako sú napríklad pre rozdelenie krajov alebo okresov.

- **Tile Layer** – nástroj, ktorý predstavuje základnú mapovú vrstvu, ktorá využíva rozhranie Leaflet Tile layer API na zobrazenie podkladových máp existujúcich poskytovateľov dlaždíc. To môže byť potrebné, keď je potrebné prepojiť údaje so skutočnými geografickými miestami.

- **Marker Layer** – nástroj, ktorý slúži na vizualizáciu údajov týkajúcich sa konkrétnych geografických miest prostredníctvom bodov (markers). Podobne ako pri choropletových polygónoch má každá značka jedinečný identifikátor a geografickú polohu (štandardne pracujeme s centroidmi krajín a práve preto používame kódy krajín).

- **Connection Layer** – nástroj, ktorý zabezpečuje vizualizáciu vzťahov medzi geopriestorovými lokalitami vo forme hrán. Užívateľ je vďaka vrstve schopný vybrať dva požadované rozmery: **od** a **do**, ktoré predstavujú uzly vykresľovaných hrán (štandardne pracujeme s centroidmi krajín identifikovanými kódmi krajín).

- **Filter Tool** – nástroj, ktorý poskytuje buď ovládacie prvky používateľského rozhrania na filtrovanie vizualizovaných záznamov údajov, alebo rozhranie API na definovanie vlastných pokročilých operácií filtrovania.
- **Selection Tool** – nástroj, ktorý poskytuje mechanizmus , ktorý prepája mapové vrstvy s vybranými geografickými objektmi mapovej vrstvy. Komunikácia medzi vrstvami je implementovaná za pomoci návrhového vzoru observer. Každá udalosť odovzdaná vrstvám obsahuje informáciu o zdrojovom prvku, ktorý vybral používateľ.
- **Themes Tool** – nástroj motívov, ktorý obsahuje sadu už dopredu preddefinovaných štýlov. Napríklad to môžu byť farby, ktoré sú poskytované iným nástrojom prostredníctvom udalostí a API, čo umožňuje definovať vlastné témy štýlov.
- **Drawing Tool** – podľa [15] je to nástroj, pri ktorom uživateľ môže zvoliť objekt za pomoci kliknutia. Zvolený objekt dokáže prispôsobovať, meniť mu výzor alebo pridať popis alebo identifikátor. Pri stlačení tlačidla myši a pohybovaní po mape počas toho ako je tlačídlo myši stlačené dochádza ku kresleniu ťahom. Spájanie polygónov nastane v prípade ak máme zvolený jeden z polygónov a počas toho vytvárame ďalší. Následne prebehne spojenie a pôvodný polygón je rozšírený o novovytvorený. Takisto je možné spojiť polygóny za pomoci nástroja, ktorý je na to určený. K rozdeleniu polygónov dochádza v momente, kedy užívateľ vytvorí polygón, je tento polygón rozdelený od všetkých polygónov pod ním, s výnimkou zvoleného. Tvorba v rámci polygónov sa môže používať pri plánoch budov. Výsledkom tejto operácie je zmena novovytvoreného polygónu, ktorý bude prienikom dvoch prvotných polygónov.



Obr. 4.1: Príklad finálneho widgetu mapy. Obsahuje bočný panel (vľavo), ktorý sa používa na konfiguráciu vrstiev, definíciu pravidiel filtrovania a všeobecné nastavenie mapy. Táto mapa obsahuje tri vrstvy: choroplet , marker a connection layer. Príklad ukazuje konfiguráciu vrstvy choropletu. Prepája dátovú doménu „od“ s vizuálnou dimenziou „krajina“, dátovú doménu „hodnota“ s vizuálnou dimenziou „hodnota“ a na agregáciu hodnôt používa funkciu „súčet“.

4.3 Definícia požiadavkov

Hlavným problémom je neexistujúce rozhranie vo forme informačného systému, ktoré by umožnilo uživateľovi využívať naplno. Vzhľadom na to boli definované následujúce požiadavky:

- Navrhnúť uživateľské rozhranie.
- Vytvoriť informačný systém, pre tvorbu geografických objektov.
- Zabezpečiť autorizáciu a autentifikáciu uživateľov.
- V rámci informačného systému, možnosť zdieľať vytvorené geoJSONY medzi uživateľmi.
- Schopnosť administrátora systému spravovať uživateľov.

Kapitola 5

Návrh

V tejto kapitole bude popísaný návrh informačného systému, na ktorom bude založená výsledná aplikácia. V rámci tejto kapitoly opíšem ako bude vyhotovený informačný systém a za pomoci akých technológií. Ďalej bude opísaná funkcionalita, ktorú bude zaobstarávať aplikácia a taktiež bude popísaný návrh užívateľského rozhrania.

5.1 Architektúra

Navrhované rozšírenie knižnice Geovisto teda spočíva vo vytvorení informačného systému. Ide o produkt, ktorý bude ľahko dostupný, pretože užívateľ bude potrebovať iba internetové pripojenie a prehliadač, pre prácu so systémom. Hlavnou cieľovou skupinou riešenia sú používatelia počítačov, a nie mobilných telefónov, z dôvodu problematiky veľkosti obrazovky. Keďže vstupy pre webovú aplikáciu sú dáta vo formáte GeoJSON a tie obsahujú rôzne definície rozličných geografických objektov. Alebo môže byť vstupom interakcia medzi užívateľom a systémom, kedy je schopný užívateľ vytvárať vlastné objekty, ktorým vie priradiť vlastné identifikátory a vďaka tomu namapovať príslušnú dimenziu na geografické objekty. Výstupom budú definované geografické objekty, ktoré budú vo formáte GeoJSON.

Užívateľ bude mať na výber sa zaregistrovať alebo v prípade už existujúceho účtu sa prihlásiť, čo mu poskytne rozšírené možnosti. Užívateľ bude mať možnosť svoje vytvorené objekty vo formáte GeoJSON si uložiť a zároveň zdieľať s inými užívateľmi, ktorí budú prihlásení do informačného systému. V prípade, že si užívateľ zvolí možnosť pokračovať do systému anonymnou cestou (neprihlási sa), bude ukrátení o spomenuté rozšírenia. Správca bude mať možnosť vytvárať, upravovať alebo mazať užívateľov.

Pre fungujúci systém bude potrebné aby bol spracovaný frontend, backend a databázu aplikácie. Systém bude obsahovať domovskú stránku, kde bude mať užívateľ horizontálnu lištu, kde bude mať na výber pokračovať ako neprihlásený užívateľ (anonym), tým sa mu ale neponúknu možnosti, aké sa ponúknu prihlásenému užívateľovi. Užívateľ bude mať možnosť si pozrieť nápovedu, ako s produktom Geovisto pracovať. Bude nutné vytvoriť prihlasovací a registrovací formulár, kde užívateľ bude pri prihlasovaní zadávať svoj email a heslo a pri registrovaní meno, email a dvakrát zopakovať heslo kvôli bezpečnosti hesla. Po prihlásení bude potrebné vytvoriť bočný panel, ktorý bude navigovať užívateľa do sekcií ako zobrazenie svojich súborov vo formáte GeoJSON, ktoré boli vytvorené daným užívateľom a takisto tie, ktoré sú s ním zdieľané. Správca bude mať možnosť vidieť zoznam prihlásených užívateľov a sledovať ním vytvorené súbory typu GeoJSON aj ktoré sú s ním zdieľané. Okrem toho bude

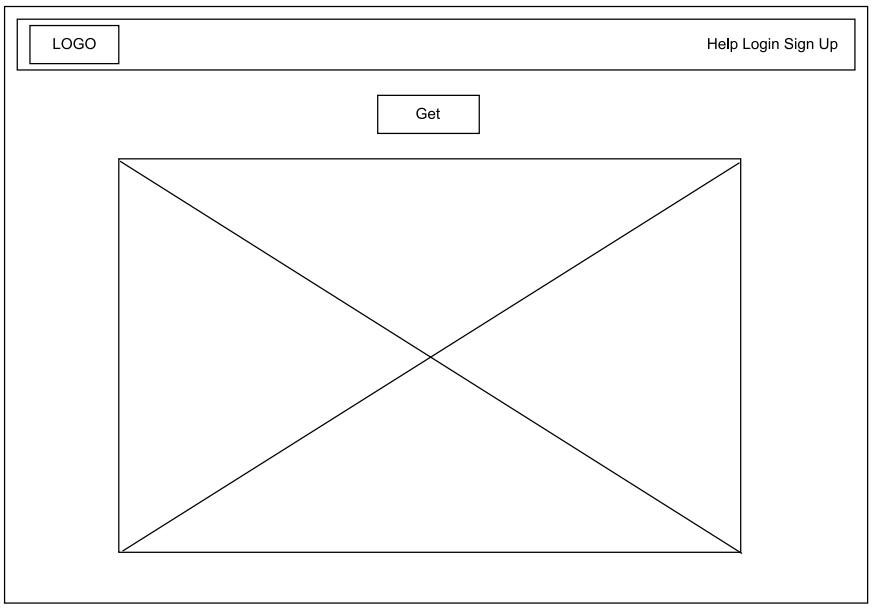
mať možnosť daných uživateľov editovať a to v prípade núdze zmeniť heslo alebo odstrániť uživateľa.

čo sa týka prihlasovania, je potrebné zabezpečiť overenie totožnosti používateľa (autentifikácia) a určenie oprávnenia používateľa (autorizácia). Bude potrebné zhotoviť tabuľku používateľov, ktorá bude slúžiť na ukladanie informácií o používateľoch systému. Informácie v tabuľke používateľov budú používané pre autentifikáciu a autorizáciu používateľov. Tabuľka používateľov môže byť implementovaná ako súčasť relačnej databázy alebo ako samostatná tabuľka v informačnom systéme. Používanie tabuľky používateľov zvyčajne zvyšuje bezpečnosť a efektivitu informačného systému, pretože sa zabezpečuje, že prístup k systému majú iba oprávnení používatelia.

5.2 Návrh pohľadov a akcií

Aplikácia bude implementovaná na základe návrhu, ktorý je spomenutý nižšie. Aplikácia bude obsahovať strany:

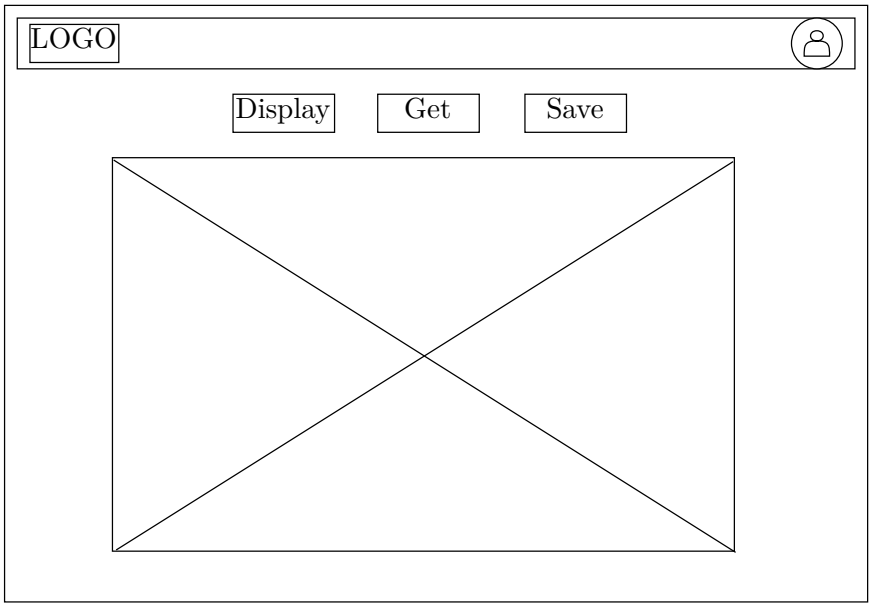
- Úvodná strana – bude obsahovať navigačnú lištu, ktorá bude implementovaná horizontálne. Bude ponúkať môžnosti help (nápoveda), home (späť na domovskú stránku), registrácia, prihlásenie sa do už existujúceho účtu alebo pokračovať ako neprihlásený (anonymný) uživateľ, viď obrázok č. *5.1*.



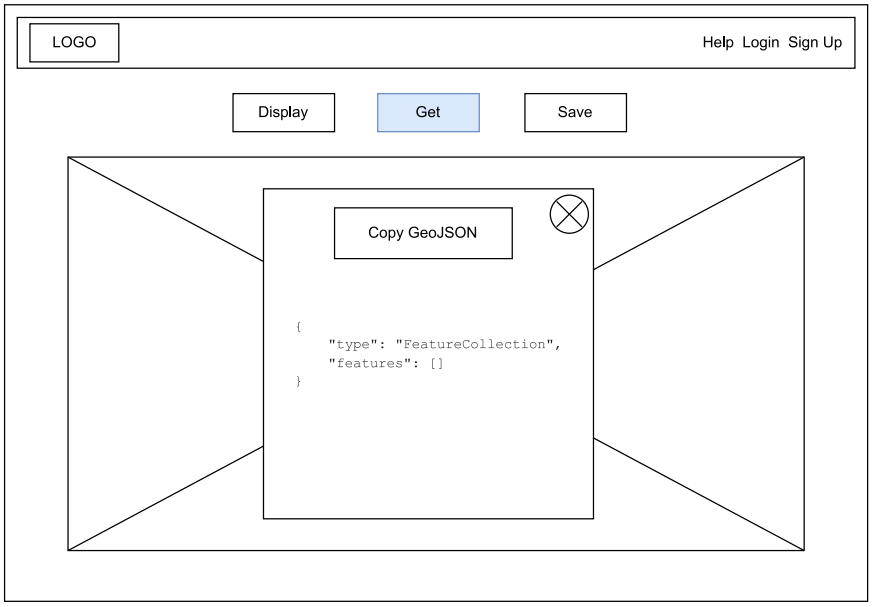
Obr. 5.1: Domovská stránka systému pre neprihláseného uživateľa

- Prihlásený uživateľ – Po prihlásení budú zobrazené na horizontálnej navigačnej lište možnosti v ľavom hornom rohu logo aplikácie, ktoré bude odkazovať na domovskú stránku a na opačnom konci navigačnej lišty bude zobrazená ikonka prihláseného uživateľa, ktorá po otvorení ponúkne možnosti prekliknutia do sekcie Geojsons, kde bude možné si prezrieť svoje uložené súbory a takisto súbory, ktoré sú s nim zdieľa iný prihlásený uživateľ. Následne je ponúknutá možnosť odhlásenie uživateľa, viď obrázok *5.2*. Príhlasený aj nepríhlasený uživateľ majú zobrazené tlačítko "Get", za pomoci

ktorého majú možnosť serializovať GeoJSON a skopírovať si daný GeoJSON, viď obrázok 5.3. Prihlásený uživateľ bude mať možnosť takisto použiť tlačítko "Save", pomocou ktorého bude schopný uložiť svoj GeoJSON do svojej tabuľky a následne tlačítko "Display", ktoré mu dokáže znovu zobraziť jeho uložený súbor.



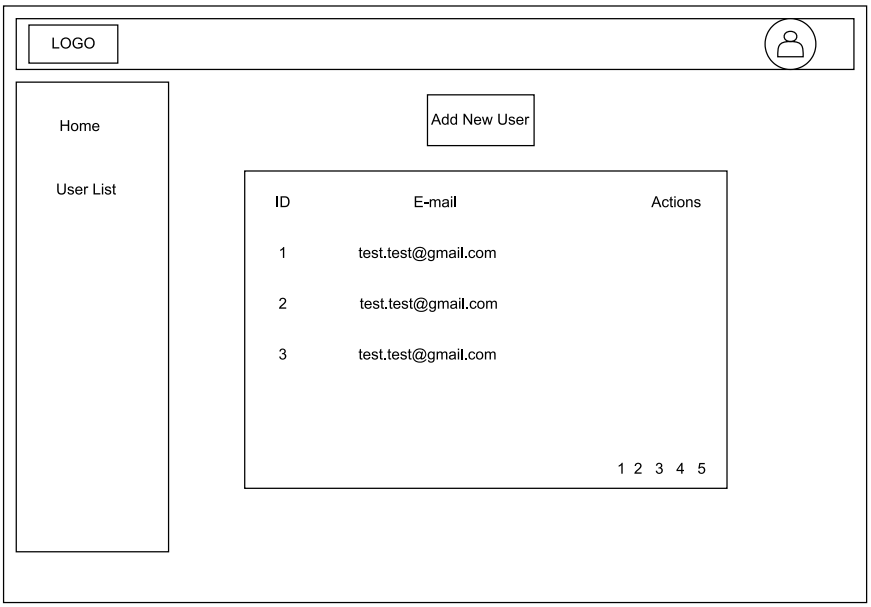
Obr. 5.2: Domovská stránka prihláseného uživateľa



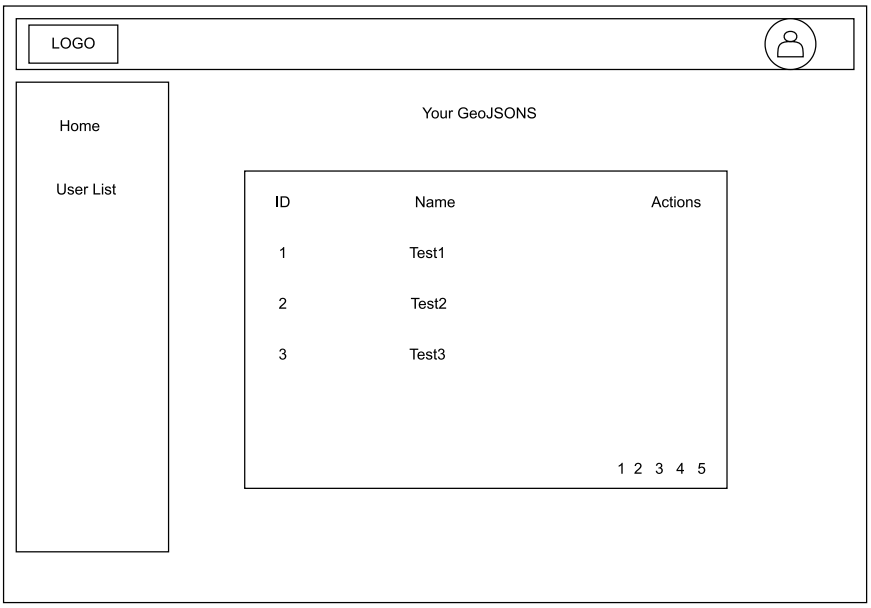
Obr. 5.3: Zobrazený GeoJSON pomocou tlačítka Get

- Správca systému po prihlásení, bude mať možnosť vidieť zobrazenie strany, ktorá obsahuje vertikálnu bočnú lištu, ktorá má na výber možnosti prekliknutia sa na domovskú stránku, zoznam uživateľov, ktorých vie následne editovať, viď obrázok 5.4. Na horizontálnej navigačnej lište bude v ľavom rohu zobrazené logo aplikácie, ktoré

bude odkazovať po kliknutí na domovskú stránku a na opačnej strane zobrazená ikonka daného uživateľa, ktorá po rozkliknutí zobrazí možnosť odhlásenia uživateľa. Po prekliknutí sa do sekcie, kde bude zobrazená tabuľka uživateľov, bude mať správca možnosť za pomoci tlačítka Ädd New User"zaregistrovať nového uživateľa. Po kliknutí na uživateľa v tabuľke, bude správcovi zobrazená tabuľka súborov, ktoré daný uživateľ vlastní alebo ktoré sú s nim zdieľané iným uživateľom, viď obrázok, *5.4*.



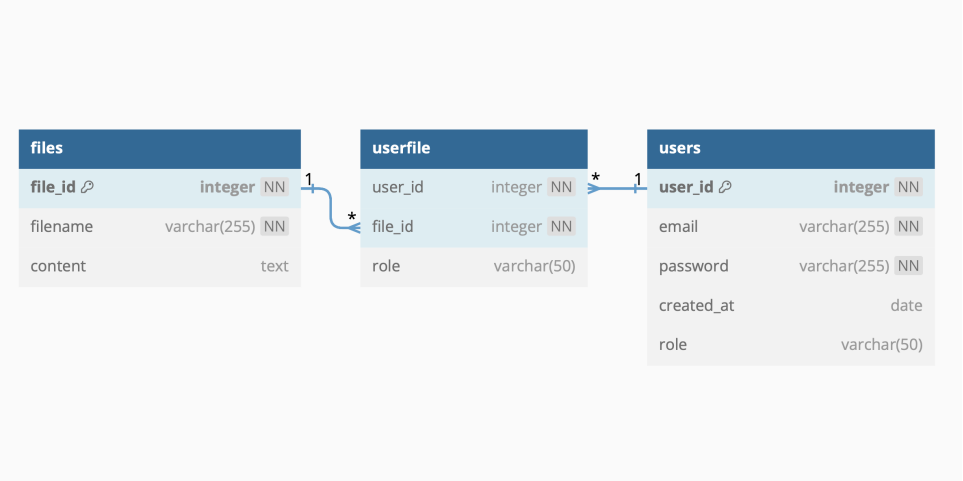
Obr. 5.4: Tabuľka uživateľov u prihláseného správcu



Obr. 5.5: Tabuľka súborov daného uživateľa u správcu

5.3 Dátový model

Dátový model reprezentuje uživateľov a ich akcie, ktoré môžu absolvovať viď obrázok *5.5*. Uživateľ bude prvotne braný ako neprihlásený. Následne ak sa prihlási alebo poprípade zaregistruje, bude vstupovať do systému ako uživateľ, o ktorom sú zaznamenávané základné informácie ako sú `meno`, `priezvisko`, `email` a `heslo`. Každého uživateľa bude identifikovať jednoznačne jeho `Uzivatel_ID`. Ak uživateľ vstupuje do systému ako rola správca, bude mať právomoci editovať uživateľov, ktorí sa zaregistrovali do systému (ich `meno`, `email` aj `heslo`). Bude schopný aj vytvárať účty do systému a takisto aj odstrániť účet uživateľa. Uživatelia budú môcť si vytvárať vlastné geoJSONy, ktoré následne si vedia uložiť a zdieľať s inými prihlásenými uživateľmi. Každý geoJSON je identifikovaný svojim primárnym kľúčom `ID_GeoJSON`. Tento identifikátor sa používa na identifikáciu geoJSONu v databáze. Z dôvodu, že každý geoJSON musí byť priradený určitému uživateľovi, bude potrebné vytvoriť vzťah medzi entitou `GeoJSON` a `Uživateľ`. Pre tento účel bude použitý cudzí kľúč. Preto bude v entite geoJSON vytvorený atribút `Uzivatel_ID`, ktorý odkazuje na primárny kľúč v entite `Uživateľ`. Medzi týmito entitami bude navrhnutý vzťah jeden ku mnohým, pretože jeden uživateľ dokáže vytvárať viacero súborov typu GeoJSON. Správca bude si môcť pozrieť zoznam svojich súborov formátu GeoJSON, ktoré daný prihlásený uživateľ vytvoril, a ktoré zdieľa s iným prihláseným uživateľom.



Obr. 5.6: Relačný model navrhovaného systému

Kapitola 6

Implementácia

Táto kapitola bude obsahovať popis implementácie informačného systému. Implementácia sa skladá z troch častí a to implementácia backendu, frontendu a databázy. Spôsob implementácie týchto častí bude v nasledujúcej sekcii postupne opísaný.

6.1 Backend

Backend je časť informačného systému, ktorá spracováva požiadavky od frontendu a zabezpečuje logiku a spracovanie dát. Pre implementáciu backendu bol použitý framework Express, postavený na Node.js. Express poskytuje jednoduché a efektívne prostredie pre tvorbu serverov a spracovanie HTTP požiadaviek.

Backend bol navrhnutý a implementovaný tak, aby poskytoval RESTful API pre komunikáciu s frontendom. Toto API umožňuje získavanie, ukladanie, aktualizovanie a mazanie dát v databáze. Express poskytuje jednoduchú a intuitívnu syntax pre definovanie routov a logiky jednotlivých koncových bodov.Overovanie užívateľa je kľúčovou súčasťou zabezpečenia systému a umožňuje riadiť prístup k chráneným častiam aplikácie.

- Generovanie JWT tokenu Prihlasovanie užívateľa začína procesom generovania JWT tokenu na serverovej strane. Po úspešnej autentifikácii užívateľa sa vygeneruje JWT token obsahujúci informácie o užívateľovi, ako napríklad identifikátor, rolu a platnosť tokenu. Tento token je následne podpísaný serverom pomocou tajomstva (secret key), aby sa zabezpečila jeho integrita a overiteľnosť.

- Odoslanie JWT tokenu frontendu Po generovaní JWT tokenu sa tento token odosiela frontendu, ktorý ho uloží na strane klienta, napríklad v lokálnom úložisku alebo v cookies. Týmto spôsobom je token dostupný pre ďalšie požiadavky na server a slúži ako prostriedok pre autentifikáciu a autorizáciu užívateľa.

- Overovanie JWT tokenu na serveri Pri každej požiadavke od frontendu je potrebné overiť platnosť a integritu JWT tokenu na serveri. Express framework poskytuje možnosť definovať middleware funkcie, ktoré sa vykonajú pred spracovaním požiadavky pre konkrétny koncový bod. Vytvoríme vlastnú middleware funkciu pre overovanie JWT tokenu.

- Implementácia middleware funkcie pre overovanie JWT tokenu Vytvorenie middleware funkcie umožňuje spracovať JWT token z príchodzej požiadavky a overiť jeho platnosť a integritu. V tejto funkcii sa vykoná nasledujúci postup:

  - Získať JWT token z príchodzej požiadavky (z hlavičky, cookies atď.).
  - Skontrolovať, či bol token poslaný a či je vo formáte JWT.
  - Overiť platnosť a integritu tokenu pomocou tajomstva (secret key).
  - Ak je token platný, získať z neho informácie o užívateľovi, ktoré sú dostupné v payload časti tokenu.
  - Tieto informácie o užívateľovi môžu byť uložené v objekte req.user alebo inom vhodnom mieste pre ďalšie spracovanie požiadavky.

- Autorizácia prístupu k chráneným častiam aplikácie Po úspešnom overení JWT tokenu a získaní informácií o užívateľovi môžeme pokračovať s autorizáciou prístupu k chráneným častiam aplikácie. Na základe rolí, oprávnení a ďalších kritérií môžeme rozhodnúť, či užívateľ má povolený prístup k danému zdroju alebo funkčnosti.

- Spracovanie chybných alebo neplatných tokenov V prípade, že prichádza požiadavka s chybným alebo neplatným JWT tokenom, je potrebné riadne spracovať túto situáciu. Môžeme vrátiť chybový stav s príslušným HTTP kódom a vhodnou chybovou správou, ktorá informuje klienta o neplatnom tokene. Taktiež je možné implementovať automatický presmerovanie na prihlasovaciu stránku alebo inú vhodnú akciu pre neplatného užívateľa.

6.2 Frontend

Frontend je časť informačného systému, ktorá sa zaoberá prezentačnou vrstvou a interakciou s používateľom. Pre implementáciu frontendu bola zvolená technológia React, populárna knižnica JavaScriptu, ktorá umožňuje vytváranie jednostránkových aplikácií. Využitie Reactu umožňuje tvorbu modulárneho a znovupoužiteľného kódu.

Pri implementácii frontendu boli vytvorené rôzne komponenty, ktoré zodpovedajú rôznym častiam užívateľského rozhrania. Tieto komponenty boli navrhnuté a implementované tak, aby poskytovali prehľadné a intuitívne používateľské rozhranie. S využitím Reactu bolo dosiahnuté dynamické načítavanie údajov a rýchla odozva aplikácie. Pri vývoji frontendu bola využitá knižnica Ant Design, ktorá poskytuje množstvo preddefinovaných komponentov a štýlov, čím zjednodušuje vývoj a zlepšuje vizuálny vzhľad aplikácie. Ant Design poskytuje rôzne komponenty, ako napríklad tlačidlá, formuláre, tabuľky, navigačné panely a mnoho ďalších. Tieto komponenty boli integrované do nášho frontendu a prispôsobené podľa potrieb informačného systému.

6.2.1 Navigačný panel

re implementáciu navigačného panelu bol použitý framework Ant Design, ktorý poskytuje množstvo preddefinovaných komponentov a štýlov. Navigačný panel je implementovaný ako samostatný komponent v React, v ktorom sú definované sekcie alebo odkazy, ktoré umožňujú používateľovi navigovať medzi rôznymi časťami aplikácie. Pri kliknutí na konkrétny odkaz sa vyvolá príslušná akcia, ako napríklad presmerovanie na inú stránku alebo zobrazenie príslušnej časti aplikácie. Navigačný panel sa líši u prihláseného a neprihláseného uživateľa. Neprihlásený uživateľ *6.1* má možnosť presunúť sa do sekcií Help alebo nápoveda, Login, pomocou ktorého sa môže prihlásiť v prípade, že je zaregistrovaný a následne Sign Up, ktorá odkazuje na už spomínané zaregistrovanie.

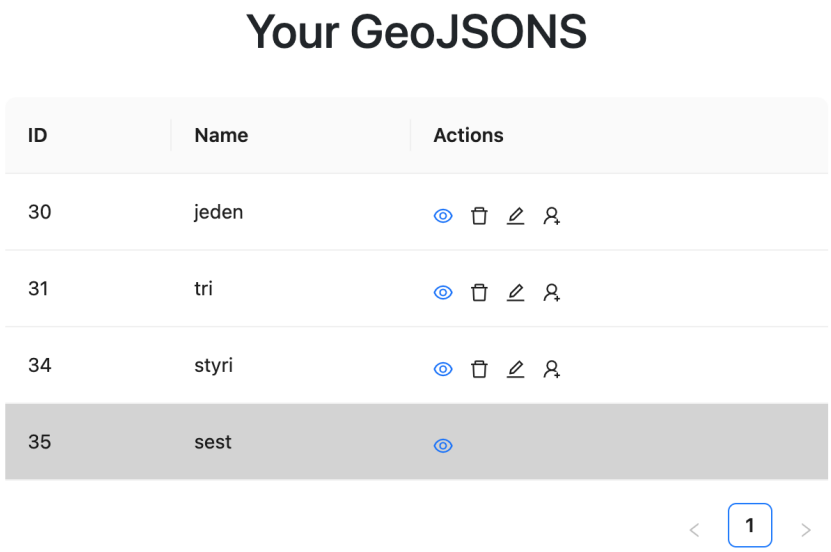
Obr. 6.1: Navigačný panel pre neprihláseného uživateľa

V prípade prihláseného uživateľa *6.2*, sa na navigačnom panely zobrazí v pravom rohu ikona avatar. Po kliknutí na danú ikonu sa objaví takzvané Dropdown Menu, ktoré bolo implementované za pomoci knižnice AntD. Odtial je uživateľovi umožnené sa presunúť do sekcie /geojsons, kde si môže pozrieť svoje uložené geojsony.

Obr. 6.2: Navigačný panel pre prihláseného uživateľa

6.2.2 Tabuľky geografických objektov

Tabuľka súborov vo formáte GeoJSON *6.3* je implementovaná pomocou knižnice Ant Design. Pre zobrazenie dát v tabuľke boli vytvorené komponenty v React, ktoré získavajú dáta zo servera. Pri načítaní stránky sa zavolá požiadavka na server, ktorá vráti zoznam súborov typu GeoJSON. Tieto dáta sa spracujú a zobrazia v tabuľke pomocou komponentov Ant Design. Používateľ môže vykonávať akcie s geojsonmi, ako sú napríklad zobraziť daný geojson, upraviť alebo odstrániť konkrétny geojson alebo ho aj zdieľať s iným uživateľom. Na začiatku boli importované potrebné závislosti, vrátane React komponentov a ikon z knižnice Ant Design. Bol vytvorený funkcionálny komponent s názvom `UserGeoJsonListTable`, ktorý obsahuje danú tabuľku. V komponente boli definované štádia pre rôzne premenné, ktoré sa budú používať, ako napríklad pre úpravu, pridanie a zobrazenie súborov, aktuálneho používateľa a ďalšie. Boli implementované funkcie pre pridanie používateľa, pridanie súboru pre používateľa, získanie súborov pre používateľa, získanie obsahu súboru, odstránenie súboru a odstránenie súboru pre konkrétneho používateľa. Následne boli vytvorené stĺpce pre tabuľku, ktoré obsahujú informácie o ID súboru, názve súboru a akciách, ako napríklad zobrazenie obsahu súboru, odstránenie súboru, úpravu súboru a pridanie používateľa. V rámci komponentu boli implementované modálne okná pre pridanie používateľa a úpravu názvu súboru. Na záver bol exportovaný komponent `UserGeoJsonListTable` pre použitie v iných častiach aplikácie.



Obr. 6.3: Tabuľka súborov typu GeoJSON daného uživateľa

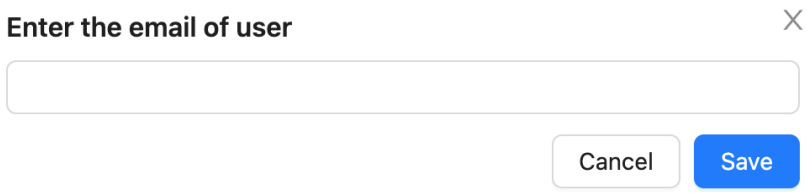
6.2.3 Ukladanie geografických objektov

Bol vytvorený komponent s názvom `PopUp` *6.4*, ktorý obsahuje tlačidlo a modálne okno pre pridanie nového súboru. V komponente boli použité stavy pre uchovávanie informácií o otvorení modálneho okna a názve súboru. V rámci modálneho okna bolo vytvorené textové pole pre zadaný názov súboru. Pri stlačení tlačidla "Save"sa volá funkcia `onAddFiles` z modulu `"../../api/auth"` s parametrami názvu súboru a obsahu súboru z props. Funkcia `onAddFiles` je definovaná v module, ktorý importuje knižnicu Axios a nastavuje predvolené konfigurácie pre Axios, aby používal cookies (vďaka axios.defaults.withCredentials = true). Samotná funkcia `onAddFiles` je asynchrónna a prijíma parameter data, ktorý obsahuje informácie o súbore, ktorý sa má pridať. Tento parameter sa odosiela ako telo požiadavky typu POST na URL `'http://localhost:8000/api/files'`. Funkcia používa knižnicu Axios na odoslanie HTTP požiadavky. Metóda axios.post vytvára požiadavku s predvolenými konfiguráciami a nastaveniami. Na základe poskytnutých parametrov (URL a data) vytvára a odosiela požiadavku na server. Požiadavka na pridanie súboru je asynchrónna, čo znamená, že funkcia vráti Promise. Výsledkom funkcie je odpoveď z backendového servera. Funkcia čaká na odpoveď z servera a potom ju vráti ako výsledok. Ak je požiadavka úspešná, výsledkom Promise bude objekt s odpoveďou, ktorý obsahuje rôzne informácie o odpovedi zo servera (napríklad stavový kód, dáta, hlavičky atď.). Ak sa vyskytne chyba, Promise bude odmietnutá a vráti sa chybový objekt.

Obr. 6.4: Popup pre uloženie geojsonu

6.2.4 Zdieľanie geografických objektov

Pri každom riadku v tabuľke súborov sa nachádza ikona `UserAddOutlined`. Po kliknutí na túto ikonu sa spúšťa funkcia `onAddUser(record)`, kde record obsahuje údaje o aktuálnom súbore. Vo funkcii `onAddUser` sa nastaví `isAdding` na `true`, čo spôsobí zobrazenie modálneho okna s názvom Ënter the email of user"*6.5*. V modálnom okne používateľ zada e-mailovú adresu užívateľa, ktorému chce pridať práva k danému súboru. E-mailová adresa je uchovaná v stave `emailString`. Po kliknutí na tlačidlo "Save"v modálnom okne sa vykoná funkcia `handleAddUser`, ktorá vykoná HTTP požiadavku typu POST na URL `'http://localhost:8000/api/get-users/files/add-user'` s e-mailovou adresou ako telom požiadavky. Backendový server spracuje požiadavku a ak je e-mailová adresa platná, pridá užívateľa s daným e-mailom k danému súboru. Ak je pridanie užívateľa úspešné, vykoná sa ďalšia akcia pomocou metódy `addUserFiles`, ktorá vykoná HTTP požiadavku na pridanie súboru užívateľovi. Akcia sa ukončí nastavením `isAdding` na `false`, čo zatvorí modálne okno.

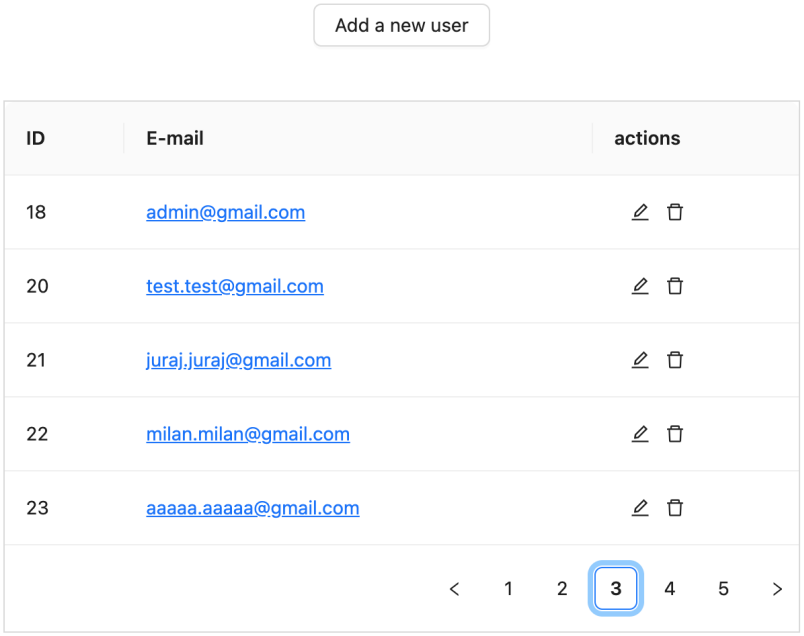


Obr. 6.5: Popup pre zdieľanie geojsonu s iným užívateľom

6.2.5 Tabuľka užívateľov

Implementácia tabuľky užívateľov `UserListTable` *6.6* zahŕňa zobrazenie existujúcich užívateľov, ich úpravu a registráciu nových užívateľov. Komponent `UserListTable` obsahuje stavové premenné pre správu stavu tabuľky, ako napríklad `isEditing` (indikuje, či sa vykonáva úprava užívateľa), `editingUser` (uchováva informácie o úprave užívateľa), `users` (zoznam existujúcich užívateľov), `isModalOpen` (indikuje, či je otvorené modálne okno pre registráciu nového užívateľa), `email` (e-mail nového užívateľa), `password` (heslo nového užívateľa), `registrationError` (chybová správa pri registrácii) a `registrationSuccess` (informácia o úspešnej registrácii). Funkcia `getUsers` sa volá pri načítaní komponentu a získava zoznam existujúcich užívateľov zo servera pomocou HTTP požiadavky typu GET na URL `'http://localhost:8000/api/get-users'`. Získané údaje sa ukladajú do stavovej premennej `users`. Pri každom riadku tabuľky je možné vykonať úpravu alebo od-

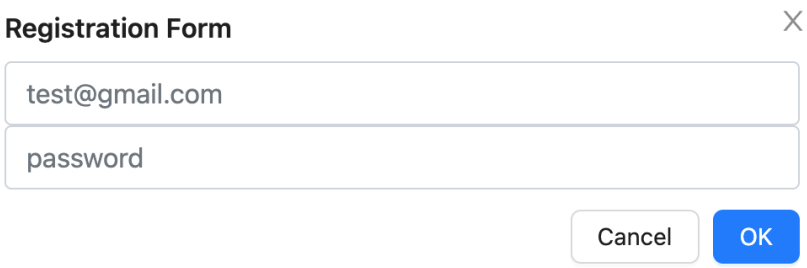
stránenie užívateľa. Tieto akcie sa vykonávajú v rámci funkcií `onEditUser(record)` a `onDeleteUser(record)`, kde record obsahuje údaje o danom užívateľovi. Funkcia `onEditUser` nastavuje stavové premenné `isEditing` a `editingUser` na príslušné hodnoty, čo spôsobuje otvorenie modálneho okna s názvom Ëdit User". V tomto okne je možné upraviť údaje o užívateľovi. Funkcia `onDeleteUser` zobrazuje modálne okno s potvrdením odstránenia užívateľa. Po potvrdení odstránenia sa aktualizuje zoznam užívateľov (odstráni sa odstránený užívateľ zo zoznamu) a pomocou HTTP požiadavky typu DELETE sa odstráni užívateľ zo servera. Po kliknutí na tlačidlo Ädd a new user"sa volá funkcia `showModal`, ktorá nastavuje stavovú premennú isModalOpen na true, čo zobrazuje modálne okno *6.7* pre registráciu nového užívateľa. Modálne okno obsahuje vstupné polia pre e-mail a heslo nového užívateľa. Po vyplnení týchto polí je možné uložiť nového užívateľa volaním funkcie handleModalSubmit. Táto funkcia vykoná HTTP požiadavku typu POST na URL `'http://localhost:8000/api/auth/register'` s údajmi o novom užívateľovi (e-mail a heslo). V prípade úspešnej registrácie sa aktualizuje stav tabuľky a zobrazí sa informácia o úspešnej registrácii. V prípade chyby sa zobrazí chybová správa. V rámci modálneho okna pre úpravu užívateľa je možné zmeniť údaje o užívateľovi (meno, e-mail, heslo). Po kliknutí na tlačidlo "Save"sa aktualizuje stav tabuľky a vykoná sa HTTP požiadavka typu PUT na URL `'http://localhost:8000/api/get-users/user_id'`, kde `user_id` je identifikátor užívateľa, ktorý sa upravuje.



Obr. 6.6: Tabuľka užívateľov

6.2.6 Registrácia užívateľa

Implementácia registrácie v komponente Register spočíva v tom, že sa využívajú React hooky `useState` na udržiavanie stavu formulára a `onRegistration` funkcia z `../api/auth` modulu pre odoslanie požiadavky na server. Po úspešnej registrácii je používateľ auto-



Obr. 6.7: Popup pre vytvorenie uživateľa

maticky prihlásený pomocou `onLogin` funkcie a nastaví sa autentifikácia v Redux store pomocou `authenticateUser` akcie z `authSlice`.

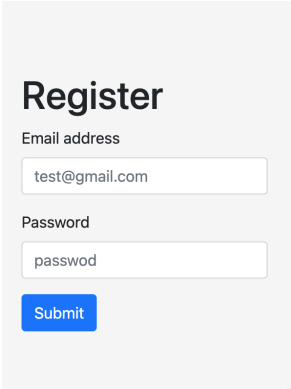
Komponent Register obsahuje nasledujúce hlavné časti:

- Deklarácia stavových premenných pomocou hookov `useState`:
  - values – Objekt, ktorý uchováva hodnoty zadané v poliach formulára (email a heslo).
  - error – Premenná, ktorá uchováva chybové hlásenie v prípade neúspešnej registrácie.
  - success – Premenná, ktorá uchováva úspešné hlásenie po registrácii.
- Vytvorenie inštancie dispatch z useDispatch hooku na manipuláciu s Redux store.
- Definovanie funkcie onChange pre zachytenie udalostí zmeny vstupov formulára. Táto funkcia aktualizuje stav values podľa zmenených hodnôt.
- Definovanie funkcie onSubmit pre spracovanie odoslaného formulára. Táto funkcia vykonáva nasledujúce kroky:
  - Zabránenie predvoleného správania formulára volaním e.preventDefault().
  - Volanie onRegistration funkcie z ../api/auth s hodnotami z values objektu.
  - Ak je registrácia úspešná, vymazanie chybového a úspešného hlásenia, vyprázdnenie polí formulára a prihlásenie používateľa volaním onLogin funkcie z ../api/auth.
  - Nastavenie autentifikácie používateľa v lokalnom úložisku (localStorage) a v Redux store volaním dispatch(authenticateUser()).
  - Ak je registrácia neúspešná, nastavenie chybového hlásenia na základe chybovej odpovede z servera.
- Vykreslenie komponentu s formulárom registrácie, ktorý obsahuje nasledujúce prvky *6.8*:
  - Vstup pre email.
  - Vstup pre heslo.
  - Chybové hlásenie v prípade neúspešnej registrácie.

– Úspešné hlásenie po úspešnej registrácii.
– Tlačidlo odoslať formulár.

Komponent Register je zabalený v komponente Layout, ktorý zabezpečuje zobrazenie základného rozloženia stránky.

Na serverovej strane je obsluha registrácie implementovaná v metóde register exportovanej z modulu. Táto metóda prijíma email a heslo z tela požiadavky, ktoré sú následne použité na vytvorenie nového používateľa v databáze. Heslo je zašifrované pomocou funkcie hash a následne je vykonaná SQL operácia INSERT na vloženie záznamu o používateľovi. Ak operácia prebehne úspešne, je vrátená odpoveď s informáciou o úspešnej registrácii. V prípade chyby pri registrácii je vrátená odpoveď so statusom 500 a informáciou o chybe.



Obr. 6.8: Registračný formulár

6.2.7 Prihlasovanie uživateľa

Prihlásenie užívateľa je implementované pomocou funkcie `Login`, ktorá je React komponentou. Vo funkcii `Login` sa používajú React hooky `useState` na uchovávanie stavu formulára. Stav obsahuje hodnoty pre email, heslo a chybovú správu.

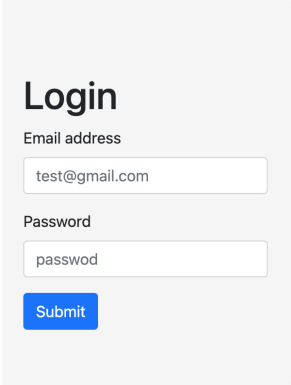
Pri zmene hodnôt vstupných polí formulára sa vyvolá funkcia `onChange`, ktorá aktualizuje stav komponenty s novými hodnotami. Pri odoslaní formulára sa vykonáva funkcia `handleSubmit`, ktorá odchytiuje udalosť `onSubmit` a vykonáva asynchrónne volanie na funkciu `onLogin` z modulu `auth`.

Ak volanie na backend prebehne úspešne, funkcia `onLogin` vráti objekt s informáciami o prihlásení. V tomto prípade sa nastavuje globálna premenná `globalEmail` s hodnotou emailu, nuluje sa chybový stav a nastavuje sa stav pre autorizáciu používateľa. Ďalej sa vykonáva volanie na backend pre získanie údajov o používateľovi na základe prihláseného emailu. Odpoveď obsahuje rolu používateľa, ktorá je uložená v lokálnom úložisku. Následne sa spúšťa Redux akcia `authenticateUser` na aktualizáciu stavu autentifikácie.

V prípade, že volanie na backend zlyhá, funkcia `onLogin` vráti chybovú odpoveď. V tomto prípade sa nastavuje chybový stav pre zobrazenie chybovej správy používateľovi.

Komponenta Login obsahuje prihlasovací formulár *6.9*, ktorý je zobrazený pomocou HTML a Bootstrap tried. Pri zmene hodnôt vstupných polí sa aktualizuje stav a vizuálna spätná väzba je poskytovaná prostredníctvom zobrazenia chybovej správy.

Celková implementácia prihlasovacieho formulára zahŕňa spracovanie zmien hodnôt, komunikáciu so serverom, správu stavu a zobrazenie výsledkov používateľovi. Funkcia `onLogin` je zodpovedná za zaslanie požiadavky na backend a spracovanie odpovede.



Obr. 6.9: Prihlasovací formulár

6.3 Databáza

Databáza je základnou súčasťou informačného systému, ktorá uchováva dáta používané aplikáciou. V tomto projekte bola pre databázový systém zvolená PostgreSQL, robustný relačný databázový systém s otvoreným zdrojovým kódom.

Databáza bola navrhnutá a implementovaná tak, aby zabezpečovala efektívne ukladanie a spracovanie dát. Boli definované potrebné tabuľky, stĺpce a vzťahy medzi nimi. S využitím SQL dotazov a transakcií je možné vykonávať rôzne operácie nad dátami, ako je vkladanie, aktualizácia, mazanie a vyhľadávanie. V tejto časti práce sa zameriame na tvorbu databázy v PostgreSQL pre informačný systém. Budeme popisovať vytvorenie troch tabuliek: `users`, `files` a `userfile`, ktoré slúžia na uchovávanie informácií o užívateľoch, súboroch a vzťahu medzi nimi.

- Tabuľka `users` slúži na uchovávanie informácií o užívateľoch v systéme. Nižšie je uvedený príklad SQL príkazu na vytvorenie tejto tabuľky:

```
CREATE TABLE public.users (
    user_id integer NOT NULL,
    email character varying(255) NOT NULL,
    password character varying(255) NOT NULL,
    created_at date DEFAULT CURRENT_DATE
);
```

  Tabuľka `users` obsahuje nasledujúce stĺpce:

  - **user_id:** Jedinečný identifikátor užívateľa (integer).
  - **email:** E-mailová adresa užívateľa (character varying).
  - **password:** Heslo užívateľa (character varying).
  - **created_at:** Dátum vytvorenia užívateľa (date). Defaultne sa nastavuje na aktuálny dátum.

- Tabuľka `file` slúži na uchovávanie informácií o súboroch v systéme. Nižšie je uvedený príklad SQL príkazu na vytvorenie tejto tabuľky:

```
CREATE TABLE public.files (
    file_id integer NOT NULL,
```

```
    filename character varying(255) NOT NULL,
    content text
);
```

Tabuľka `files` obsahuje nasledujúce stĺpce:

- **file_id:** Jedinečný identifikátor súboru (integer).
- **filename:** Názov súboru (character varying).
- **content:** Obsah súboru (text).

- Tabuľka `userfile` slúži na zachytenie vzťahu medzi užívateľmi a súbormi v systéme. Nižšie je uvedený príklad SQL príkazu na vytvorenie tejto tabuľky:

```
CREATE TABLE public.userfile (
    user_id integer NOT NULL,
    file_id integer NOT NULL,
    role character varying(50)
);
```

Tabuľka `userfile` obsahuje nasledujúce stĺpce:

- **user_id** Identifikátor užívateľa (integer).
- **file_id:** Identifikátor súboru (integer).
- **role:** Rola alebo povolenie pre užívateľa vzhľadom na daný súbor (character varying).

Týmto spôsobom sme vytvorili tabuľky users, files a userfile, ktoré slúžia ako základné entity pre ukladanie informácií o užívateľoch, súboroch a vzťahoch medzi nimi v informačnom systéme implementovanom v PostgreSQL databáze. Tieto tabuľky poskytujú štruktúru a schému pre ukladanie a manipuláciu s dátami v systéme.

Kapitola 7

Testovanie

V tejto kapitole sa budeme venovať testovaniu informačného systému pre tvorbu a správu geografických objektov. Testovanie je kritickým procesom pri vývoji softvéru, pretože nám umožňuje identifikovať a odstrániť chyby a nedostatky, zabezpečuje funkčnosť, spoľahlivosť a bezpečnosť systému a zaručuje, že systém bude plniť očakávania a požiadavky užívateľov. V tejto kapitole sa zameriame na rôzne typy testovania, testovacie scenáre, nástroje na testovanie a výsledky testovania nášho informačného systému.

7.1 Testovanie systému užívateľmi

Okrem môjho vlastného testovania, som systém ponúkol na vyskúšanie aj reálnym osobám rôznych vekových kategórií a profesionalného zamerania, aby bolo testovanie najefektívnejšie. Testovacie osoby si daný systém vyskúšali, a zaznamenali svoje dotazy, ktoré som následne čítal a reflektoval na ne.

7.1.1 Testovací adept 1

Ide o 25 ročného študenta, ktorý študuje na Fakulte infromačných technológií v Prahe. Počas testovania sa zameral na funkčnosť systému z hľadiska tvorby nových geografických objektov a správy existujúcich. Testoval rôzne typy geografických dát, vrátane polygonov, línií a bodov.

Výsledky testovacieho adepta 1:

- Tvorba nového geografického objektu prebehla hladko. Adept bol spokojný s intuitívnym užívateľským rozhraním, ktoré mu umožnilo jednoducho pridať nový polygon na digitálnu mapu.
- Správa existujúcich objektov bola bezproblémová. Uživateľ rýchlo našiel všetky svoje vytvorené objekty.

7.1.2 Testovací adept 2

Ide o 40 ročného softvérového inžiniera, ktorý sa zaoberá testovaním softvéru. Jeho cieľom bolo overiť výkonnosť systému, ako aj jeho bezpečnosť. Skúmal, ako dobre systém zvláda zaťaženie pri spracovaní požiadavkov a či sú zabezpečené užívateľské údaje.

Výsledky testovacieho adepta 2:

- Pri testovaní výkonnosti systému bol uživateľ spokojný s rýchlosťou odpovedí a spracovaním požiadavkov.
- Testovanie bezpečnosti odhalilo niekoľko menších bezpečnostných nedostatkov, ktoré boli však rýchlo identifikované a opravené v rámci testovacieho procesu.

7.1.3 Testovací adept 3

Ide o 25 ročnú študentku geografie, ktorá sa zaujíma o geografickú analýzu a vizualizáciu dát. Počas testovania sa venovala najmä použiteľnosti a vizuálnemu spracovaniu systému. Skúmala, ako dobre sa orientuje v užívateľskom rozhraní, či je pre ňu prirodzené vykonávať rôzne operácie a navigovať v systéme.

Výsledky testovacieho adepta 3:

- Uživateľka považovala užívateľské rozhranie za veľmi prehľadné a intuitívne. Rýchlo sa naučila, ako vykonávať základné úlohy, ako je pridávanie nových bodov a znovunačítanie existujúcich objektov.

7.1.4 Návrh ďalšieho rozšírenia

Aj keď navrhovaný informačný systém poskytuje užívateľom významné možnosti práce s geografickými objektmi a poskytuje efektívne nástroje na správu a zdieľanie dát vo formáte GeoJSON, existujú stále možnosti pre ďalšie rozšírenia a vylepšenia systému. Nasledujúce návrhy na ďalšie rozšírenia majú za cieľ zvýšiť funkcionalitu a prispôsobiteľnosť informačného systému, čo by zlepšilo používateľský zážitok a rozšírilo možnosti jeho využitia:

- S cieľom umožniť užívateľom jednoduchú zdieľanie a ukladanie dát, mohlo by byť užitočné integrovať informačný systém s cloudovými úložiskami, ako sú Google Drive, Dropbox alebo iné. Týmto by sme uľahčili užívateľom prístup k ich dátam z rôznych zariadení a umožnili im jednoduché zdieľanie s ostatnými používateľmi.
- Okrem GeoJSON môže byť užívateľom užitočné pracovať s inými formátmi geografických dát, ako napríklad KML (Keyhole Markup Language) alebo Shapefile. Rozšírením systému o podporu ďalších formátov by sme zvýšili interoperabilitu a umožnili užívateľom pracovať s rôznymi typmi geografických dát bez potreby ich konverzie.

Kapitola 8

Záver

Cieľom tejto semestrálnej práce bolo navrhnúť a implementovať informačný systém pre správu geografických objektov, ktoré budú vytvorené pomocou programovej knižnice Geovisto. V rámci práce sme sa zameriavali na vytvorenie užívateľsky prívetivého a efektívneho systému, ktorý umožní užívateľom jednoduché a prehľadné pracovanie s geografickými dátami, ako aj ich ukladanie, zdieľanie a importovanie.

V prvej časti práce sme sa venovali základným pojmom týkajúcim sa geografických dát a ich spracovania. Predstavili sme geografický informačný systém (GIS) a jeho využitie v rôznych odvetviach. Ďalej sme analyzovali spôsoby práce s geografickými dátami, ako je zobrazenie vektorových a rastrových dát, a spomenuli sme aj možnosti ich spracovania a analýzy.

V druhej časti sme sa zameriavali na vizualizáciu geografických dát a predstavili sme programové knižnice Leaflet, MapBox a OpenLayers, ktoré nám umožnili vytvoriť interaktívne mapové aplikácie. Taktiež sme zhodnotili rôzne webové frameworky pre tvorbu užívateľských rozhraní, ktoré sme použili pre implementáciu nášho informačného systému.

V časti analýzy sme sa venovali potrebám a požiadavkám užívateľov a skúmali sme súčasný stav existujúcich mapových nástrojov. Na základe tejto analýzy sme stanovili konkrétne požiadavky, ktoré informačný systém mal spĺňať, aby bol čo najviac prispôsobený potrebám užívateľov.

V návrhovej časti sme detailne rozpracovali architektúru systému, navrhli sme pohľady a akcie, ktoré užívateľom umožňovali jednoduchú manipuláciu s geografickými dátami, a definovali sme dátový model. Tieto kroky nám pomohli vytvoriť prehľadný a dobre štruktúrovaný systém.

Pri implementácii sme sa sústredili na tvorbu backendu a frontendu nášho informačného systému. Implementovali sme navigačný panel, tabuľky pre geoJSONy, mechanizmy ukladania a zdieľania dát, ako aj funkcionalitu registrácie a prihlásenia užívateľov. Systém sme úspešne podporili pomocou objektovo-relačného databázového systému PostgreSQL.

Výsledkom našej práce je funkčný informačný systém pre správu geografických objektov, ktorý poskytuje užívateľom možnosť efektívne pracovať s geografickými dátami. Systém je jednoduchý na používanie, ponúka interaktívne mapové nástroje a umožňuje užívateľom ukladať, zdieľať a importovať vlastné geoJSONy. Veríme, že navrhnutý systém bude cenným nástrojom pre rôzne odvetvia, ktoré pracujú s geografickými dátami a pomôže im efektívne analyzovať a vizualizovať priestorové informácie.

Literatúra

[1] Azaz L. *The use of Geographic Information Systems (GIS) in Business* [online]. 2011. Dostupné z: http://mlsvc01-prod.s3.amazonaws.com/f27e9ca7001/7290b7e9-0fb4-4394-9817-b2234970b27b.pdf.

[2] Brown, D. G., Elmes, G., Kemp, K. K., Macey, S. a Mark, D. *Geographic information systems. Geography in America at the Dawn of the 21st Century GL Gaile, CJ Willmott (Eds.)(Oxford University Press, Oxford)*. 2004, s. 353–375.

[3] Butler H., Hobu Inc., Daly M., Cadcorp, A. Doyle, Gillies S., Hagen S., Schaub T. *The GeoJSON Format*. 2016. Dostupné z: https://www.rfc-editor.org/rfc/rfc7946.txt.

[4] Cadenas C. *Geovisualization: Integration and Visualization of Multiple Datasets Using Mapbox*. Dostupné z: https://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=1140&context=cpesp.

[5] Cihelský L.,Valentová V. *Význam základní klasifikace ukazatelů pro korektní interpretaci vzájemných odlišností jejich hodnot*. 2006. Dostupné z: https://polek.vse.cz/pdfs/pol/2006/04/06.pdf.

[6] Dent D. B., Torgusson J. S., Hodler T. H. *Thematic Map Design* [online]. McGraw-Hill, 2009. Dostupné z: https://handoutset.com/wp-content/uploads/2022/06/Cartography-Thematic-Map-Design-Borden-Dent-Thomas-Hodler-Jeff-Torguson.pdf.

[7] Environmental Systems Research Institute. *ESRI Shapefile Technical Description*. 1998. Dostupné z: https://www.esri.com/content/dam/esrisites/sitecore-archive/Files/Pdfs/library/whitepapers/pdfs/shapefile.pdf.

[8] GISGeography. *Vector vs Raster: What's the Difference Between GIS Spatial Data Types?* Dostupné z: https://gisgeography.com/spatial-data-types-vector-raster/.

[9] Hynek J., Kachlík J., Rusňák V. *Geovisto: A Toolkit for Generic Geospatial Data Visualization*. Dostupné z: https://www.scitepress.org/Papers/2021/102604/102604.pdf.

[10] Hynek J., Rusňák V. *Towards Interactive Geovisualization Authoring Toolkit for Industry Use Cases*.

[11] Kraak M.J., Ormeling F. *Visualization of Geospatial Data*.

[12] Lichner V. *Základy štatistiky v sociálnych vedách I.* [online]. 2020. Dostupné z: https://unibook.upjs.sk/img/cms/2020/ff/zaklady-statistiky-v-socialnych-vedach.pdf.

[13] Neene V.,Kabemba M. *Development of a Mobile GIS Property Mapping Application using Mobile Cloud Computing.* 2017. Dostupné z: https://pdfs.semanticscholar.org/6a53/32f6e213953d69f6465286f8eabe87008efb.pdf.

[14] Overflow, S. *2021 Developer Survey* [online]. 2022. Dostupné z: https://survey.stackoverflow.co/2022/#most-popular-technologies-webframe.

[15] Tlčina A. *Webová aplikácia pre definíciu grafických objektov na mape* [online]. 2020. Dostupné z: https://www.fit.vut.cz/study/thesis-file/23887/23887.pdf.

[16] Wernecke J. *TEXKML Handbook - Geographic Visualization for the Web.* Google, 2009. ISBN 0-321-52559-0.