

Projekt – K155UZPR



GeoLog

Josef Jehlička

Uživatelské prostředí pro záznam turisticky
navštívených obcí

Cíle práce:

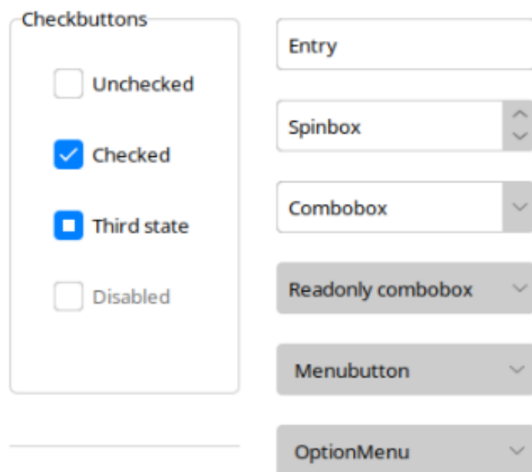


- Vytvořit uživatelské prostředí
- Zobrazit polygony obcí, okresů a krajů
- Zobrazovat jednotlivé kraje a okresy
- Vytvořit lokální databázi
- Tvořit, mazat a vybírat uživatele jako tabulky
- Manuálně přidávat a mazat obce
- Přidávat obce ze záznamu trasy (Mapy.cz)
- Vypisovat obce dle data navštívení
- Vytvořit statistiku navštívenosti krajů a okresů
- Obarvovat navštívené obce červeně
- Kompilovat aplikaci do .exe
- Vytvořit uživatelskou příručku

Tvorba uživatelského prostředí



- **Tkinter** – open-source python modul pro Tk GUI toolkit



- Okna, tlačítka, rozbalovací menu, textová okna, posuvníky

- Předefinovaná **tcl stylizace** objektů
<https://github.com/rdbende/Azure-ttk-theme/>



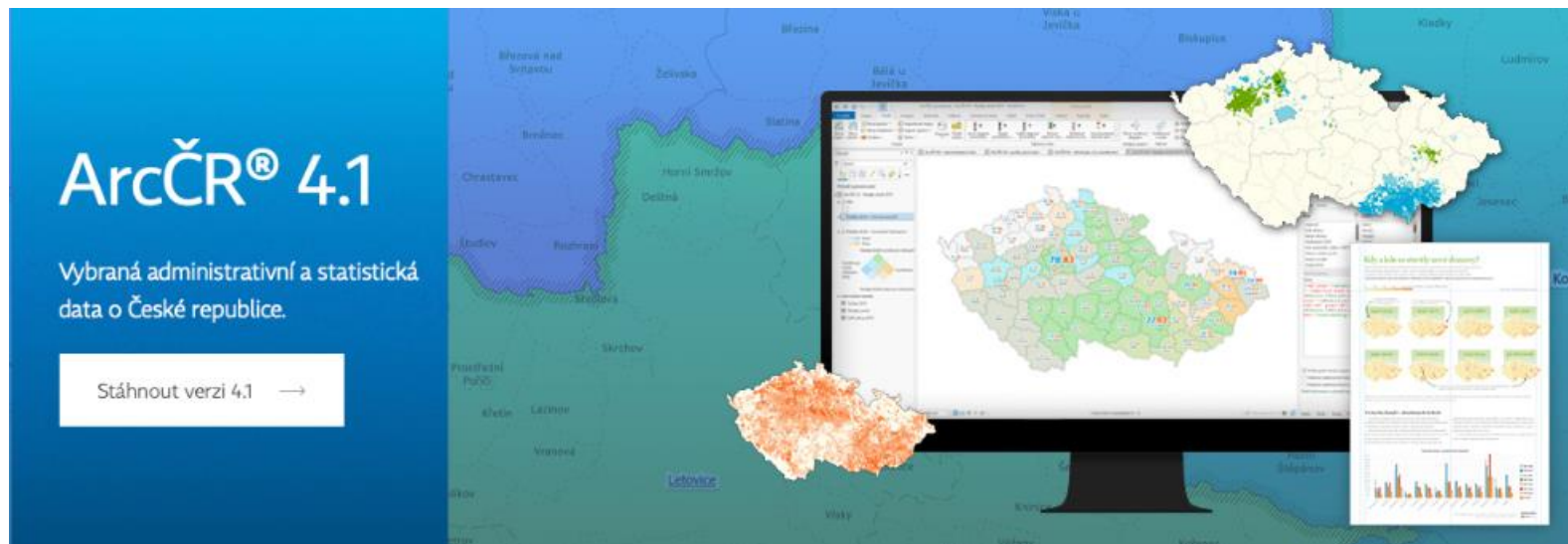
Ukázka Python skriptu

Tkinter

```
root = tk.Tk()
root.geometry(f"{{(screen_width - 100)}}x{{(screen_height - 100)}}")
root.title("GeoLog")
root.iconbitmap("files/ico.ico")
root.tk.call("source", "files/azure.tcl")
root.tk.call("set_theme", "light")
root.resizable(False, False)

if screen_width / screen_height > 1.6:
    date_picker_root.geometry(f"{{(int(screen_width * 0.30))}}x{{(int(screen_height * 0.45))}}")
else:
    date_picker_root.geometry(f"{{(int(screen_width * 0.40))}}x{{(int(screen_height * 0.55))}}")
```

Tvorba polygonů



- Polygony územních jednotek byly získány z **ArcČR 4.1**
- Bylo nutné využít atributy geometrie, názvu obce, okresu, kraje a kód obce,

- Polygony obcí byly generalizovány v softwaru **QGIS** nástrojem v.generalize (**GRASS**)
- Uloženy jako **ShapeFile**



Zobrazování polygonů

matplotlib

- **from** matplotlib.backends.backend_tkagg **import** FigureCanvasTkAgg



Přibliž na kraj:

--vyber kraj--

Přibliž na okres:

--vyber okres--

- Použito pro načítání a filtrování dat z ShapeFile
- Obce jsou filtrovány dle volby kraje/okresu v rozbalovacím menu



Ukázka Python skriptu

```
for gpkg_path, color in gpkg_paths:
    # Read GeoPackage file using GeoPandas
    gdf = gpd.read_file(gpkg_path)

    # Plot only the borders of the GeoDataFrame on Matplotlib axis and color them
    gdf.boundary.plot(ax=ax, color=color)

    # Create a FigureCanvasTkAgg
    canvas = FigureCanvasTkAgg(fig, master=root)
    canvas_widget = canvas.get_tk_widget()
    canvas_widget.pack(side=tk.TOP, fill=tk.BOTH, expand=1)

# Update the canvas after both plots
canvas.draw()
root.update()

# Destroy the loading window once the shapefiles are loaded
loading_window.destroy()
```

matplotlib

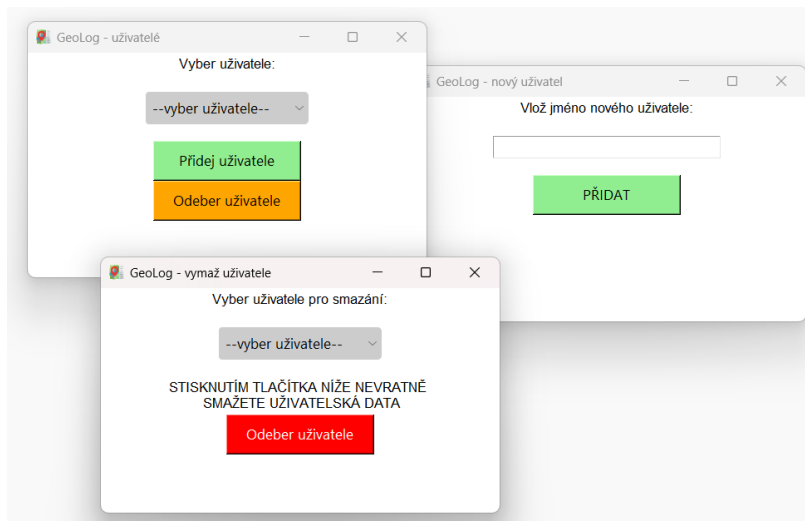
Tvorba databáze uživatelů



- Lokální databáze **SQLite3**
- Uživatelé jsou tvořeni jako jednotlivé tabulky

- V tabulce se pro každou obec ukládá její číslo v číselníku obcí (CISOB) a datum navštívení

Column Name	Type	Nullable
id	INTEGER	NO
obecID	VARCHAR(6)	NO
dat	DATE	NO



- Byla vytvořena Tkinter okna pro tvorbu, mazání a výběr existujících uživatelů pomocí SQL příkazů

SQLite3

Manuální přidávání a mazání obcí



GeoLog - přidej obec

Vyber kraj:

Pardubický kraj

Vyber okres:

Pardubice

Vyber obec:

Pardubice

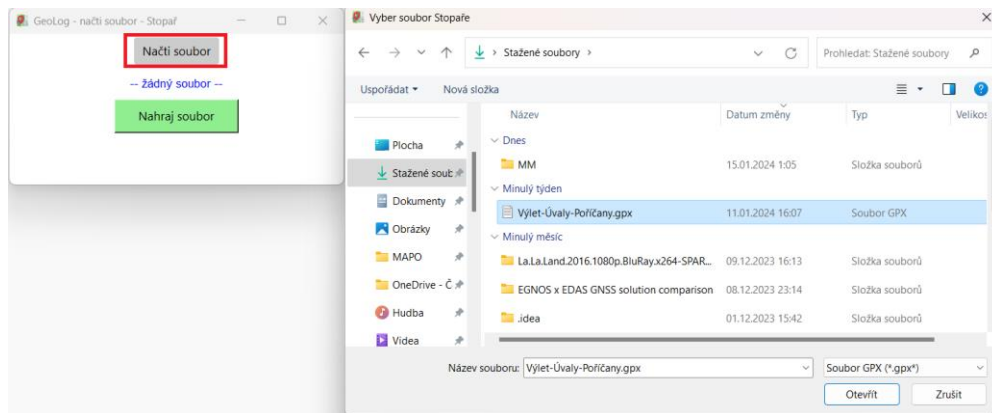
Vyber datum:

24-01-2024

Ledna 2024

	po	út	st	čt	pá	so	ne
1	1	2	3	4	5	6	7
2	8	9	10	11	12	13	14
3	15	16	17	18	19	20	21
4	22	23	24	25	26	27	28
5	29	30	31	1	2	3	4
6	5	6	7	8	9	10	11

Nahrávání souboru GPX (Mapy.cz)



- Bylo vytvořeno okno pro načítání **GPX** souboru

- Soubor je parserován pomocí knihovny **gpxpy**
- Zeměpisné šířky a délky každého 10tého bodu jsou uloženy
- Body jsou transformovány z WGS84 do S-JTSK (**pyproj**)
- Získají se čísla obcí, ve kterých leží alespoň 1 bod (**GeoPandas**)
- Pro každou obec se získá datum jejího prvního výskytu v GPX souboru
- Obce s daty jsou vloženy do databáze



Ukázka Python skriptu

gpxpy

```
n = 10 # Every n-th point will be used

# Iterate through tracks, segments, and points
for track in gpx.tracks:
    for segment in track.segments:
        filtered_points = segment.points[::n]
        latitudes.extend(point.latitude for point in filtered_points)
        longitudes.extend(point.longitude for point in filtered_points)

# Convert latitude and longitude to the S-JTSK coordinate system
transformer = Transformer.from_crs("EPSG:4326", "EPSG:5514")
coordinates = transformer.transform(latitudes, longitudes)

# Select ObecID based on the coordinates
selected_obecIDs = []
for i in range(len(coordinates[0])):
    try:
        selected_obecIDs.append(
            obce_shp[obce_shp['geometry'].contains(Point(coordinates[0][i], coordinates[1][i]))][
                'kod_obce'].iloc[0])
    except:
        print("bod mimo ČR")
        continue
```

Filtrace obcí dle data

GeoLog - filtrování dle data

Vyber mezi kterými daty
chceš filtrovat své obce:

OD:

12-01-2023

DO:

23-01-2025

Filtruj

13-12-2023 - Poříčany (Kolín)

13-12-2023 - Klučov (Kolín)

13-12-2023 - Úvaly (Praha-východ)

13-12-2023 - Český Brod (Kolín)

13-12-2023 - Tuklaty (Kolín)

13-12-2023 - Rostoklaty (Kolín)

22-01-2024 - Stárvkov (Náchod)

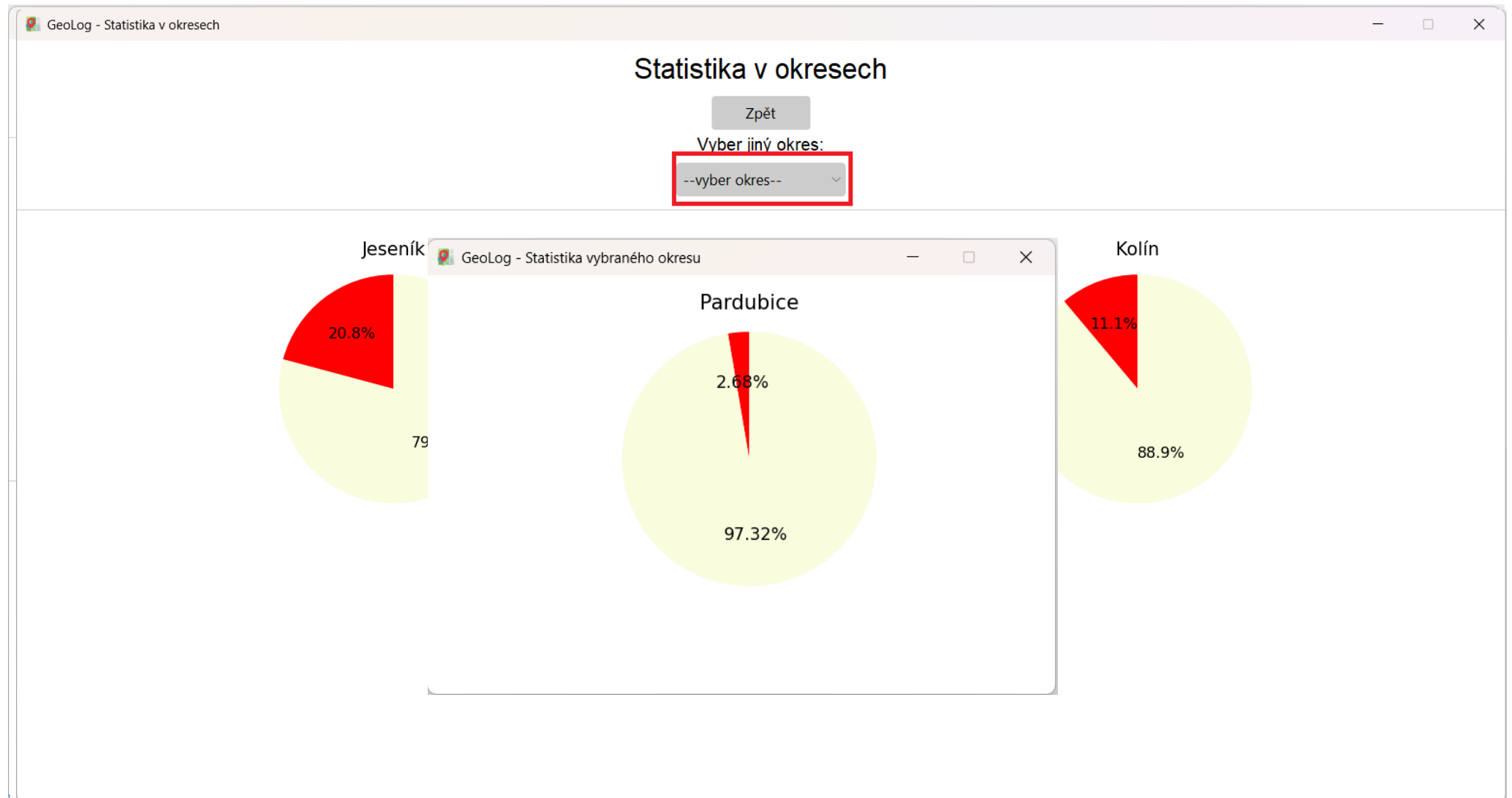
22-01-2024 - Horní Planá (Český Krumlov)

22-01-2024 - Sloveč (Nymburk)

22-01-2024 - Hroznová Lhota (Hodonín)



Statistika



Ukázka Python skriptu

matplotlib

```
okr_obecIDs = obce_shp[obce_shp['nazev_okre'] == okr]['kod_obce'].unique()

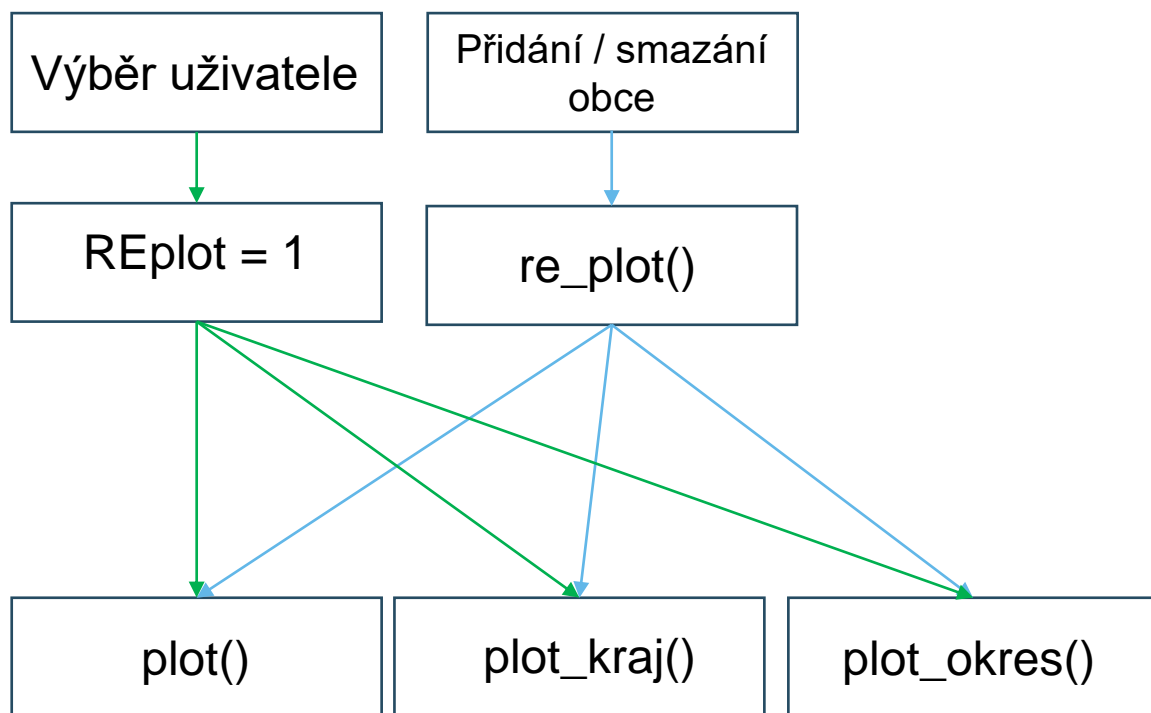
# Pick the obecIDs from the database that are in the selected okres
okr_obecIDs_db = cursor.execute(
    f"SELECT obecID FROM {user} WHERE obecID IN ({','.join(map(str, okr_obecIDs))})")
okr_obecIDs_db = okr_obecIDs_db.fetchall()
percentage = round((len(okr_obecIDs_db) / len(okr_obecIDs)) * 100, 2)

def create_pie_chart(value, title):
    colors = ['red', '#F9FCDD']
    fig, ax = plt.subplots()
    percentages = [value, 100 - value]
    ax.pie(percentages, autopct='%1.2f%%', startangle=90, colors=colors)
    ax.axis('equal')
    ax.set_title(title)

    return fig

print("Okres:", okr)
print("Procento navštívenosti[%]:" , percentage)
fig = create_pie_chart(percentage, f"{okr}")
```


Obarvování obcí v grafu



- Při výběru uživatele se globální proměnná REplot přepíše na 1
- Při změně počtu obcí se spustí funkce re_plot()
- Podle aktuálního přiblížení se spustí daná vykreslovací funkce
- Pokud je ve vykreslovací funkci splněna podmínka REplot == 1, vyberou se obce z databáze a jsou nabarveny červeně.



Ukázka Python skriptu

replot()

```
def re_plot():

    global REplot

    # get current zoom
    kraj_select = combo_var_kraje.get()
    okres_select = combo_var_okresy.get()

    # plot visited obce in database based on the selected value in the combobox (current zoom)
    if kraj_select == "--vyber kraj--" and okres_select == "--vyber okres--":
        plot_geopackage(root, gpkg_paths[::-1], loading_window)

    elif kraj_select != "--vyber kraj--" and okres_select == "--vyber okres--":
        loading_window.destroy()
        plot_geopackage_selection(root, gpkg_paths_no_kraje[::-1], loading_window, kraj_select)

    elif kraj_select != "--vyber kraj--" and okres_select != "--vyber okres--":
        loading_window.destroy()
        plot_geopackage_selection_okr(root, gpkg_paths_no_kraje[::-1], loading_window, okres_select)

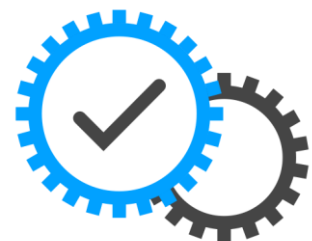
    else:
        print("chyba pri prekresleni")
```

Kompilace & Tvorba příručky



- Kompilace proběhla pomocí knihovny **PyInstaller**
- Skript byl kompilován do jednoho .exe souboru s nastavenou ikonou

- Uživatelská příručka byla napsána v **LaTeXu** na platformě **Overleaf**
- Popisuje ovládání aplikace a způsob získání GPX souboru z mobilní aplikace Mapy.cz





DĚKUJI
ZA POZORNOST