ZüriWieNeu – Spatial Analysis of reported urban issues in the City of Zurich

This project is concerned with urban issues in Zurich, reported by the population and provided by the ZüriWieNeu platform. It allows inhabitants to report different sorts of issues in the city. They can provide a detailed description of the problem and put it into a specific problem category, e.g. Abfall/Sammelstelle, Grünflächen/Spielplätze, Strasse/Trottoir/Platz, and more. While this is a great opportunity for the population to involve in optimizing their living area and directly mention their problem, without long, complicated bureaucratic procedures, as well as for the city, to simplify the identification and resolving of current issues, it lacks an important step, when it comes to making the most out of that valuable data, so that measures can be implemented most efficiently. To do so, we need to structure and analyse the data in a meaningful way, to be able to answer specific questions that might help resolve these issues. Generally speaking, those questions ask how the reported issues are spatially distributed and in what numbers, which problems are dominant in which areas, are there reports more frustrated and therefore more urgent than others and how has the amount changed over time. The specific questions are the following four:

1. *Which Quartiers receive the highest number of reports?*
2. *Which problem-categories dominate in which Quartier?*
3. *What is the proportion of frustrated Reports in each Quartier, and how are the frustrated Reports clustered over the city?*
4. *How has the number of Reports changed over time, and in which Quartiers is the Rate of Change significantly higher?*

With that, that ultimate aim of this analysis is, to analyse the spatial characteristics of the reported issues, in order to provide insight into the data and a fact-based foundation, that allows for a more efficient problem resolution and a better resource allocation. That way, we not only have a lot of data, but we are actually able to make direct use of it, and what it tells us.

Workflow and Implementation

To reach the overarching goal of this analysis and answer the specific questions, a structured workflow is necessary. Summarized, it starts with loading and inspecting the raw data, cleaning the raw data, preparing it for answering the specific questions, by structuring and organizing it and performing spatial, as well as non-spatial analysis, to answer the questions in a next step and adding meaningful visualisations, and finally summarizing the findings and providing an interactive map, including the insights from all questions.

The first crucial step is loading and inspecting the raw data. While it sounds simple, already two important steps are done here, in addition to looking at the actual content and the datatypes of the data. One, the CRS gets automatically checked and transformed to the correct target CRS needed in this specific analysis. Second, all unnecessary columns get dropped, to get rid of distracting unmeaningful information, save storage, and make the further steps faster.

The second step, which is the data preparation and analysis, might be the most important one. While it doesn't directly answer the questions, it provides all the necessary information for it. At first, all needed metrics and new fields, that can be calculated without a spatial join are generated and added to the data. Then, a spatial join is done, by performing a point in polygon analysis, linking the unique reports to their according Quartier. Based on that, all remaining new fields and metrics are calculated and added to the right (geo)data frame. These parts build the foundation of the further analysis and allows answering the specific question.

In the next step, all questions are answered. With the prepared data from the previous section, meaningful plots and maps are generated, that visualize the answers. They consist of plots (bar, or line plots) and maps, that belong together, provide the necessary context and directly show, what is needed to answer the questions. At the end of this section, the final, processed data are stored, including all newly generated field, and also including a few insightful fields from the raw data, that has not been dropped, despite it was not yet used in the analysis, because it might be needed for future questions. That said, the aim was not only to answer the questions, but also to provide new, enriched data, that can be locally saved to use for the readers own purpose, if interested. In addition, all the plot static maps are also saved.

Last but not least, a final section provides a short overview of the findings, as well as an interactive map, including the information of the previous static maps and some extra information. This is not the main part of the analysis, but rather an extra, so that the data can be further explored in an interactive way.

Results

The analysis clearly shows trends and patterns. When it comes to the raw report count, some Quartiers stand out, having mor reports that the rest. The top 5 in descending order are: Langstrasse, Sihlfeld, Altstetten, Unterstrass, Wipkingen. However, if we normalize the raw numbers with the area of the Quartiers to get the reports per km2 of each quartier, the picture changes. Then, the top 5 in descending order are: Langstrasse, Werd, Lindenhof, Rathaus, Sihlfeld. So, while for the raw count, the top candidates were found mainly in Quartiers with the biggest area, the ones with the highest report density are all very much concentrated around the city centre and the more build up, densely populated areas (Fig. 1).

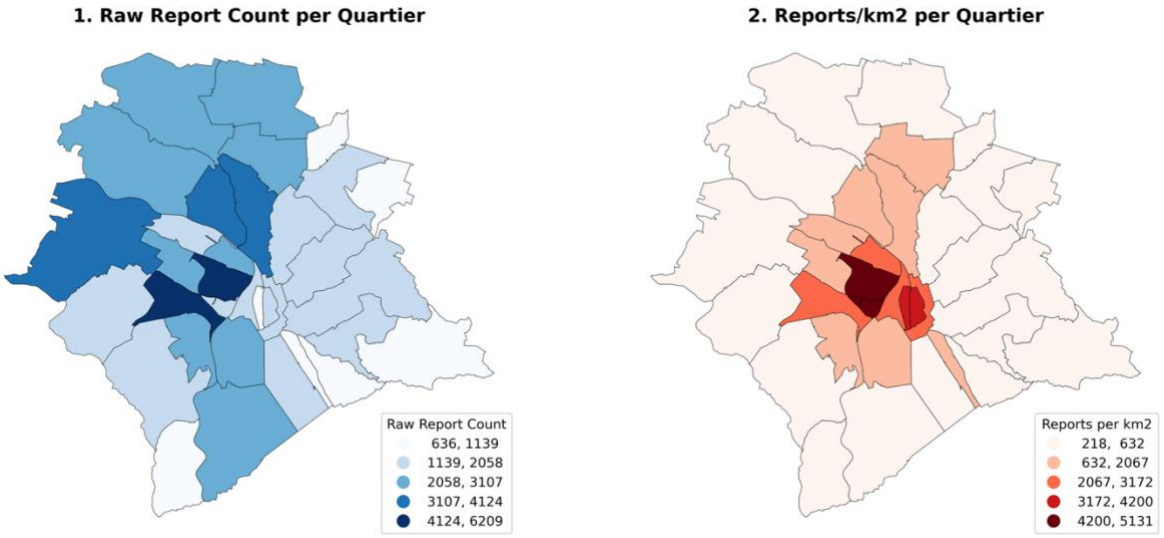


Figure 1. Chorpleth map comparing the raw, unnormalized report count and the area_normalized report count per Quartier.

The distribution of the top problem category also varies. But the top 3 categories are almost the same for all Quartiers, being Abfall/Sammelstelle, Strasse/Trottoir/Platz, and Signalisation/Lichtsignal (Fig. 2). When looking at the spatial distribution of only the most dominant Problem, all Quartiers have Abfall/Sammelstelle, except 3. For Escher Wyss and City, it is Strasse/Trottoir/Platz, and for Friesenberg it is Grünflächen/Spielplätze.

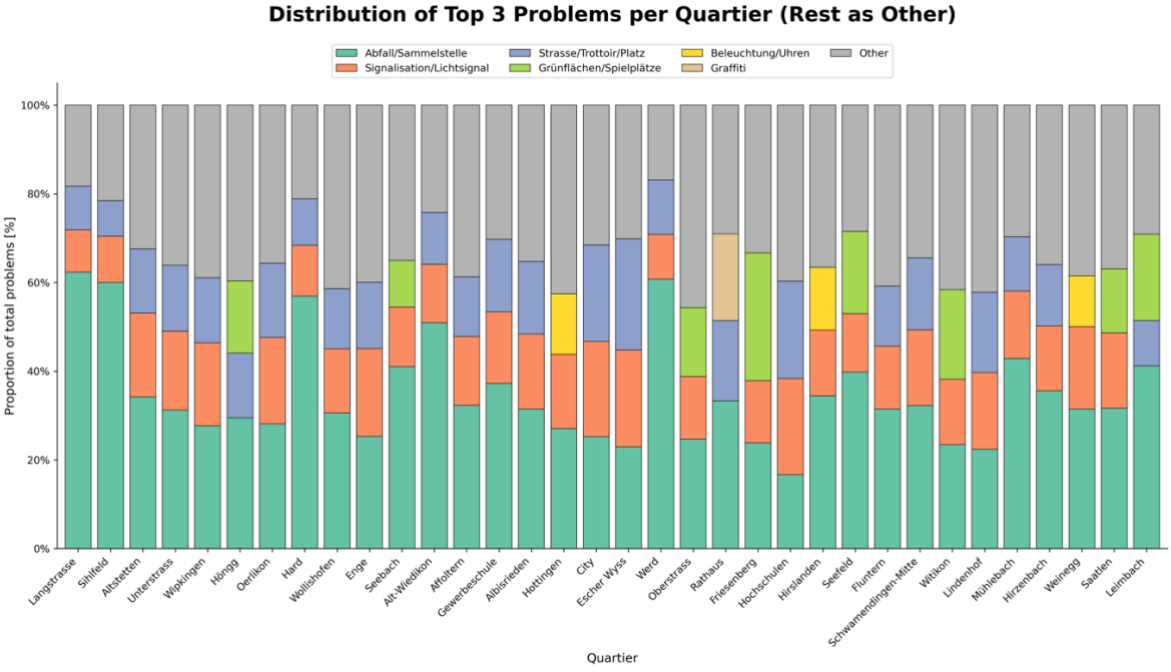


Figure 2. Distibution of top 3 problems and the rest together as "other", for each Quartier.

Looking at the frustration of the reports and the spatial distribution, it gets a bit more complicated. While Witikon, Schwammendingen-Mitte, Escher Wyss and Saatlen stand out with over 20%, the aggregated data to the Quartiers on the map does not show a trend of concentration towards the centre. But the DBSCAN Clusters show more Hotspots in the centre (Fig. 3/4). This might be because one compares the frustration between the Quartiers, and the other just looks at the reports over all and identified clusters of frustrated reports. But since there are more reports concentrated in the centre, the chance of finding clusters of frustrated reports there is higher, already because auf the overall higher density.

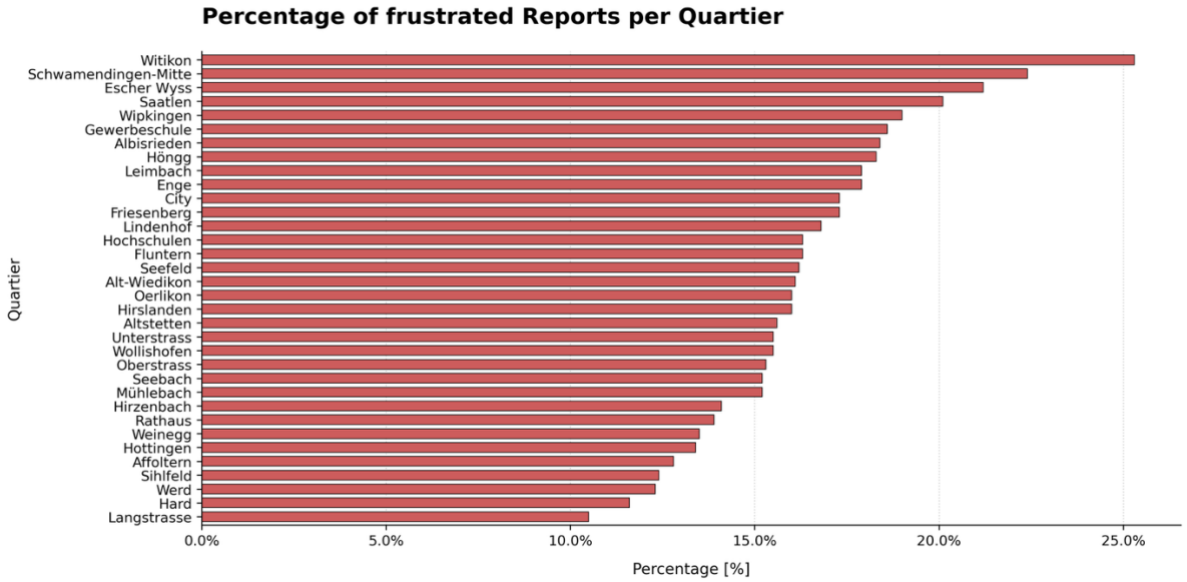


Figure 3. Horizontal barplot, showing the percentage of frustrated reports per Quartier.

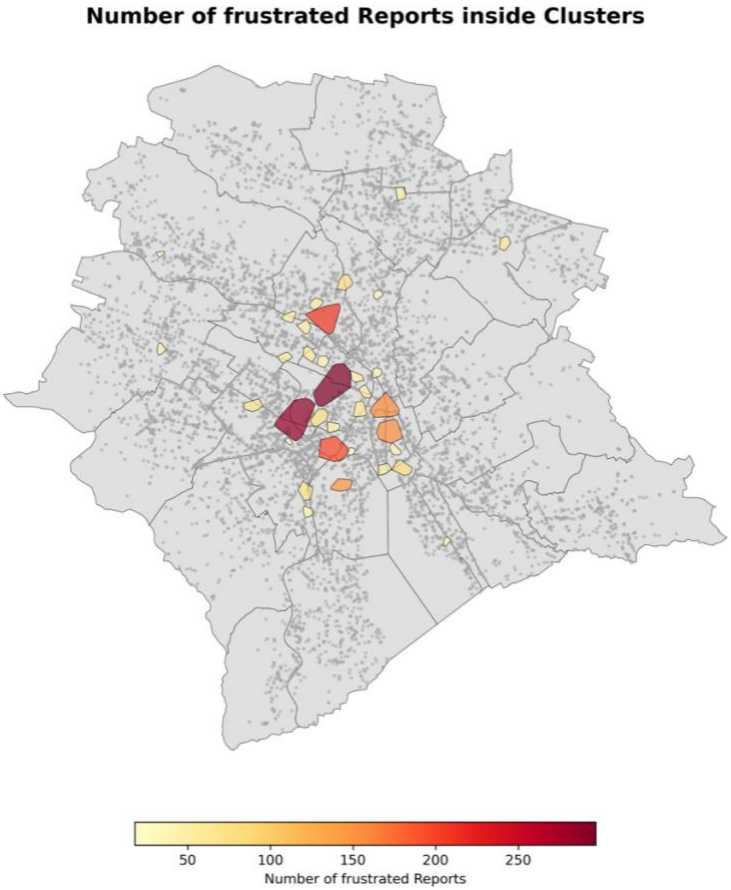


Figure 4. Map showing the cluster polygons, coloured by the number of reports, belonging to a cluster.

Lastly, when looking at the temporal trend and the growth rate, we can observe, that before 2019 (2013- 2018), the yearly number of reported issues is relatively constant, and after 2019 (2020- 2025), the number have more than doubled, with 2019 being a bit on an outlier, or better said 2020, which has lower number than 2019 (Fig. 5). This might be due to COVID19, but this is just a speculation. When looking at the growth rate, Langstrtasse clearly stand out with the highest one, being 383% (first period compared to the second, 2013 – 2018 vs. 2020-2025). Second and third are Sihlfeld (338%) and Seefeld (299%). But similar to the frustration rate, those with a high growth rate are not only concentrated in the centre (Fig. 6).

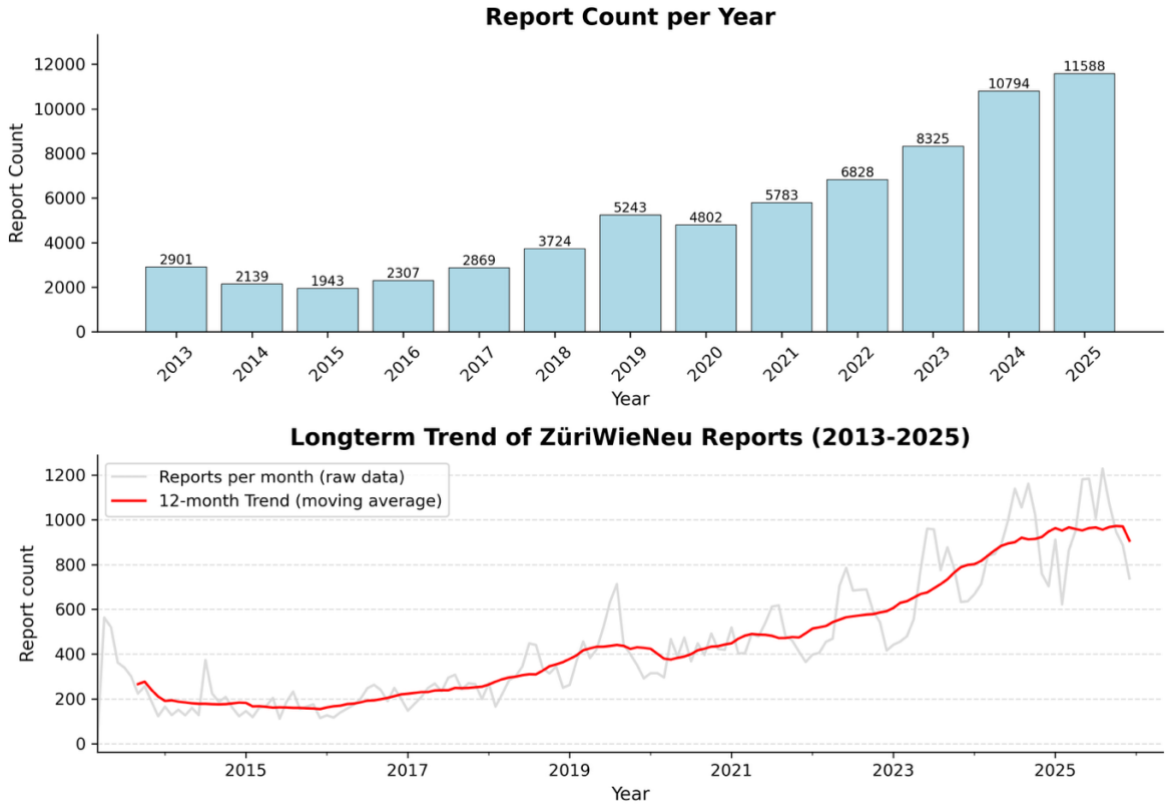


Figure 5. Longterm trend (2013 - 2025) for the development of the report counts per year.

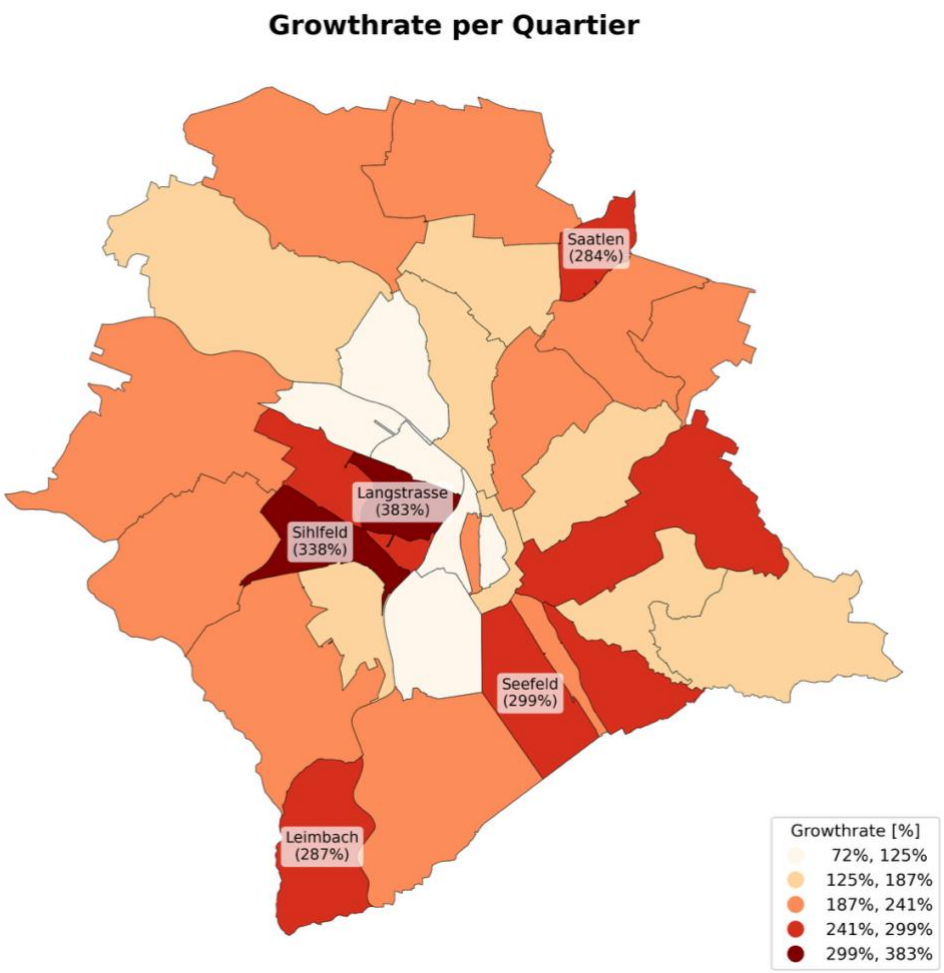


Figure 6. Chorpleth map, showing the growthrate in percent for each Quartier. The top 5 are labeled with their name and the specific growthrate.

Challenges and Reflection

Although everything worked out in the end, there have been some challenges. One example was the establishment of a logical workflow and finding a good structure of the notebook. This was even hardened when it came to outsourcing some parts into function into the utils.py. To deal with that, patience was needed and cautiously going through the whole notebook many times. Towards the end, it was also important to go from experimenting and getting to know as much as possible from the data, to really thinking about what the notebook should tell in the end, and therefore deciding, what structure makes the most sense and what of the content is needed. A more analytical type of challenge was the definition of the 4 questions and deciding what to do with the data ultimately. To deal with that, it was necessary to stay rather simple, trying not to overcomplicate it, e.g. by using machine learning for flagging frustrated reports.

In the end, this analysis provides good, basic insight and shows that, the reported urban issues underly a clear spatial distribution. But there are many more things to analyse, and even more work has to be done, adding relevant context to the findings. Only this would allow for a proper, fact-based interpretation, and not some false conclusions.

Link to GitHub-Repository: https://github.com/mlengenfelder/SDS210_ZuriWieNeu

AI Statement

I used a LLM as an assistance tool during the preparation of this project to check and help debug code for data processing. In addition, it was used for optimising the interactive map, regarding additional elements that have not been thought in the course. Apart from the uses listed above, no AI tools were used in the preparation of this submission, and all final logic and writing was verified by me.

ZüriWieNeu – Spatial Analysis of reported urban issues in Zurich

Author: Maximilian Lengenfelder

Project Description: *This project structures and analyzes urban issue reports from the "ZüriWieNeu" platform in Zurich (2013–2025). By applying spatial joins and clustering algorithms (DBSCAN), the analysis answers questions regarding report density, dominant problem categories, spatial hotspots of frustrated reports, and long-term reporting trends.*

0. Instructions

Before starting this notebook, please read the **README.md!**

You can run this notebook step by step (which is recommended), or as a whole. As also mentioned in the README.md, make sure to create your environment, according to the *environment.yml*! Afterwards, don't forget to select the *Kernel* in the notebook (Kernel name).

1. Setup and Configuration

- Import all libraries (Pandas, GeoPandas, etc.)
- Define relative paths

In [1]:
```
%load_ext autoreload
%autoreload 2

import sys
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import matplotlib.ticker as mtick
import folium
from folium.plugins import HeatMap
from pathlib import Path
from folium.plugins import MarkerCluster

ROOT_DIR = Path.cwd().parent

if str(ROOT_DIR) not in sys.path:
    sys.path.append(str(ROOT_DIR))

from src import utils as ut

PATH_RAW = ROOT_DIR / 'data' / 'raw'
PATH_REPORTS   = ROOT_DIR / 'data' / 'raw' / 'Reports_GPKG' / 'data' / 'data.gpkg'
PATH_QUARTIERE = ROOT_DIR / 'data' / 'raw' / 'Quartiere_ZH_GPKG' / 'data' / 'data.gpk
PATH_OUTPUTS   = ROOT_DIR / 'outputs'
PATH_PROCESSED_DATA = ROOT_DIR / 'data' / 'processed'

PATH_RAW.mkdir(parents=True, exist_ok=True)
PATH_OUTPUTS.mkdir(parents=True, exist_ok=True)
```

```
PATH_PROCESSED_DATA.mkdir(parents=True, exist_ok=True)
(PATH_OUTPUTS / 'figures').mkdir(parents=True, exist_ok=True)
(PATH_OUTPUTS / 'data').mkdir(parents=True, exist_ok=True)

print('Environment ready: Libraries loaded and paths set!')
```

```
Environment ready: Libraries loaded and paths set!
```

2. Data Loading & Cleaning

2.1 Load the Reports and Quartiere, inspect the raw data to get an overview and assert the CRS!

In [2]:
```
raw_reports_gdf = gpd.read_file(PATH_REPORTS, engine='pyogrio')
raw_quartiere_gdf = gpd.read_file(PATH_QUARTIERE, layer='stzh.adm_statistische_quarti

raw_reports_gdf   = ut.check_and_fix_crs(raw_reports_gdf, 'raw_reports_gdf')
raw_quartiere_gdf = ut.check_and_fix_crs(raw_quartiere_gdf, 'raw_quartiere_gdf')

assert raw_reports_gdf.crs == raw_quartiere_gdf.crs == 'EPSG:2056', 'ERROR: CRS-Trans
```

```
raw_reports_gdf is already in Target CRS: EPSG:2056
raw_quartiere_gdf is already in Target CRS: EPSG:2056
```

2.2 And drop unnecessary columns!

In [3]:
```
DROPLIST_REPORTS = [
    'objectid',
    'e',
    'n',
    'service_code',
    'userid',
    'title',
    'media_url',
    'service_notice',
    'url',
    'description'
]

DROPLIST_QUARTIERE = [
    'objid',
    'kname',
    'knr'
]

raw_reports_gdf = raw_reports_gdf.drop(columns=DROPLIST_REPORTS, errors='ignore')
raw_quartiere_gdf = raw_quartiere_gdf.drop(columns=DROPLIST_QUARTIERE, errors='ignore

display(raw_reports_gdf.head(2))
display(raw_quartiere_gdf.head(2))

display(raw_reports_gdf.info())
display(raw_quartiere_gdf.info())
```

	service_request_id	requested_datetime	agency_sent_datetime	updated_datetime	service
0	1	2013-03-14 15:16:15	2013-04-04 07:25:05	2013-04-12 07:59:30	Strasse/Trott
1	2	2013-03-14 15:17:57	2013-03-26 14:05:05	2013-04-12 08:00:22	Strasse/Trott

	objectid	qname	qnr	geometry
0	1	Alt-Wiedikon	31	POLYGON ((2680606.662 1247034.584, 2680626.356...
1	2	Witikon	74	POLYGON ((2685858.632 1246502.629, 2685860.738...

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 73343 entries, 0 to 73342
Data columns (total 9 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   service_request_id    73343 non-null  str
 1   requested_datetime    73343 non-null  datetime64[ms]
 2   agency_sent_datetime  72508 non-null  datetime64[ms]
 3   updated_datetime      73343 non-null  datetime64[ms]
 4   service_name          73343 non-null  str
 5   status                73343 non-null  str
 6   detail                73343 non-null  str
 7   interface_used        73343 non-null  str
 8   geometry              73343 non-null  geometry
dtypes: datetime64[ms](3), geometry(1), str(5)
memory usage: 5.0 MB
None
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 34 entries, 0 to 33
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   objectid  34 non-null     int32
 1   qname     34 non-null     str
 2   qnr       34 non-null     int32
 3   geometry  34 non-null     geometry
dtypes: geometry(1), int32(2), str(1)
memory usage: 948.0 bytes
None
```

3. Data Preparation

- Calculate new basic fields for Reports (**month**, **year**, **solving_duration**, **is_frustrated**) and for Quartiere (**area_km2**)
- Spatial Join: Perform a Points in Polygon analysis, linking the Reports to the according Quartier
- Calculate new advanced fields after spatial join, to create the final, complete GDF's, with all metric needed for answering the Queastions Q1 - Q4!

3.1 Calculate new basic fields for Reports & Quartiere

In [4]:
```
FRUSTRATION_KEYWORDS = [
    'gefährlich', 'gefahr', 'dringend', 'sofort', 'unakzeptabel',
    'ärgerlich', 'ärger', 'empört', 'empörend', 'wütend', 'unzumutbar',
    'skandal', 'inakzeptabel', 'schon lange', 'immer noch', 'schon wieder',
```

```
    'seit wochen', 'seit monaten', 'bereits', 'wiederholt', 'mehrmals',
    'nichts passiert', 'keine reaktion', 'ignoriert', 'trotz', 'unglaublich',
    'gefährdet', 'verletzung', 'verletzt',
]

base_reports_gdf = ut.prepare_reports_base(raw_reports_gdf, FRUSTRATION_KEYWORDS)
base_quartiere_gdf = ut.prepare_quartiere_base(raw_quartiere_gdf)

display(base_reports_gdf[base_reports_gdf['is_frustrated']].head(2))
display(base_quartiere_gdf.head(2))
```

	service_request_id	requested_datetime	agency_sent_datetime	updated_datetime	servic
16	19	2013-03-22 15:46:05	2013-04-24 10:45:05	2013-04-24 16:24:12	Strasse/Trot
33	39	2013-04-05 20:27:48	2013-04-05 20:30:06	2013-04-16 07:13:06	Beleuchtu

	objectid	qname	qnr	geometry	area_km2
0	1	Alt-Wiedikon	31	POLYGON ((2680606.662 1247034.584, 2680626.356...	1.69
1	2	Witikon	74	POLYGON ((2685858.632 1246502.629, 2685860.738...	4.93

3.2 Spatially join Reports & Quartiere, and create new advanced fields for answering the Questions (Q1 - Q4)

In [5]:
```
PERIOD_1 = list(range(2013, 2019))
PERIOD_2 = list(range(2020, 2026))

final_reports_gdf, final_quartiere_gdf, category_count, growth_df = ut.sjoin_and_stat

rates = final_quartiere_gdf['growth_rate'].dropna()
min_rate, max_rate = rates.min(), rates.max()

if min_rate < 0:
    print('Colourramp-Test: Negative Growthrates found -> Using centered TwoSlopeNorm
    vmin = min(min_rate, -0.1)
    vmax = max(max_rate, 0.1)
    norm = mcolors.TwoSlopeNorm(vmin=vmin, vcenter=0.0, vmax=vmax)
    cmap_to_use = 'coolwarm'
else:
    print('Colourramp-Test: Only positive Growthrates found -> Using linear Normalize
    vmin = 0.0
    vmax = max(max_rate, 0.1)
    norm = mcolors.Normalize(vmin=vmin, vmax=vmax)
    cmap_to_use = 'OrRd'

display(final_reports_gdf.head(2))
display(final_quartiere_gdf.head(2))
```

```
Number of unassigned points: 6
Unassigned points were successfully dropped to 0!
Colourramp-Test: Only positive Growthrates found -> Using linear Normalize!
```

	service_request_id	requested_datetime	agency_sent_datetime	updated_datetime	service
0	1	2013-03-14 15:16:15	2013-04-04 07:25:05	2013-04-12 07:59:30	Strasse/Trott
1	2	2013-03-14 15:17:57	2013-03-26 14:05:05	2013-04-12 08:00:22	Strasse/Trott

	objectid	qname	qnr	geometry	area_km2	report_count	avg_solving_days	frustration_
0	1	Alt-Wiedikon	31	POLYGON ((2680606.662 1247034.584, 2680626.356...	1.69	2535	3.1	
1	2	Witikon	74	POLYGON ((2685858.632 1246502.629, 2685860.738...	4.93	1145	3.9	

4. Analysis: Answering four specific Questions

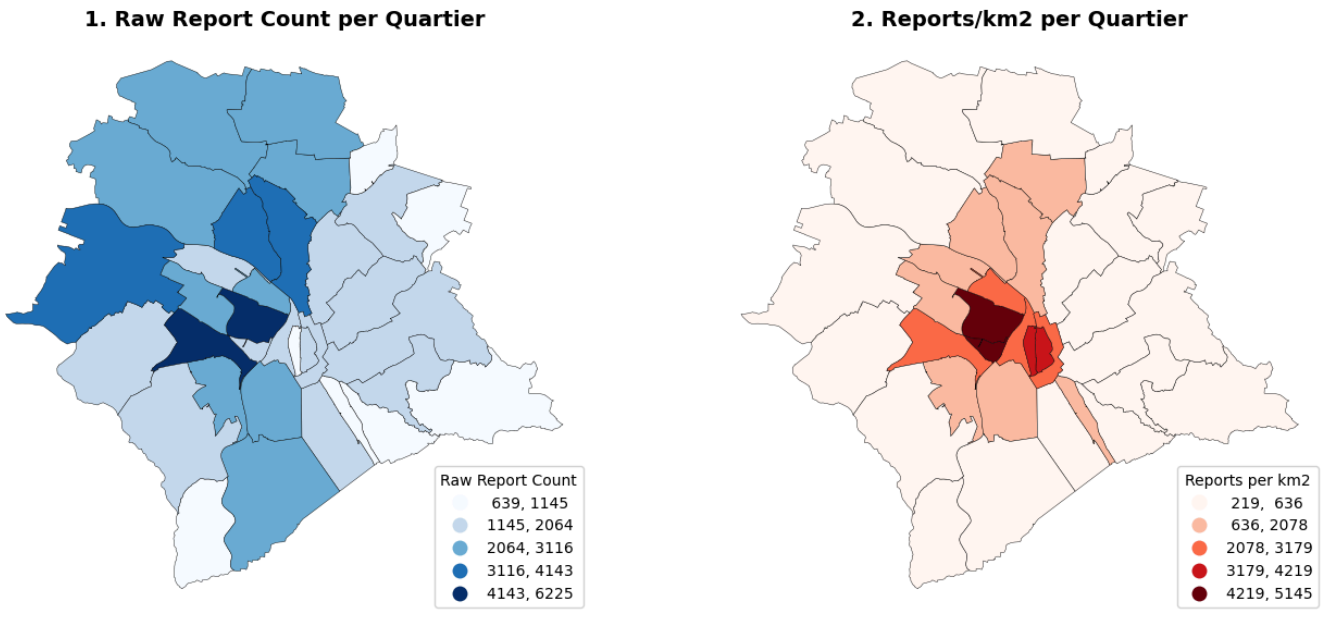
4.1 **Q1: Report Count and Density**

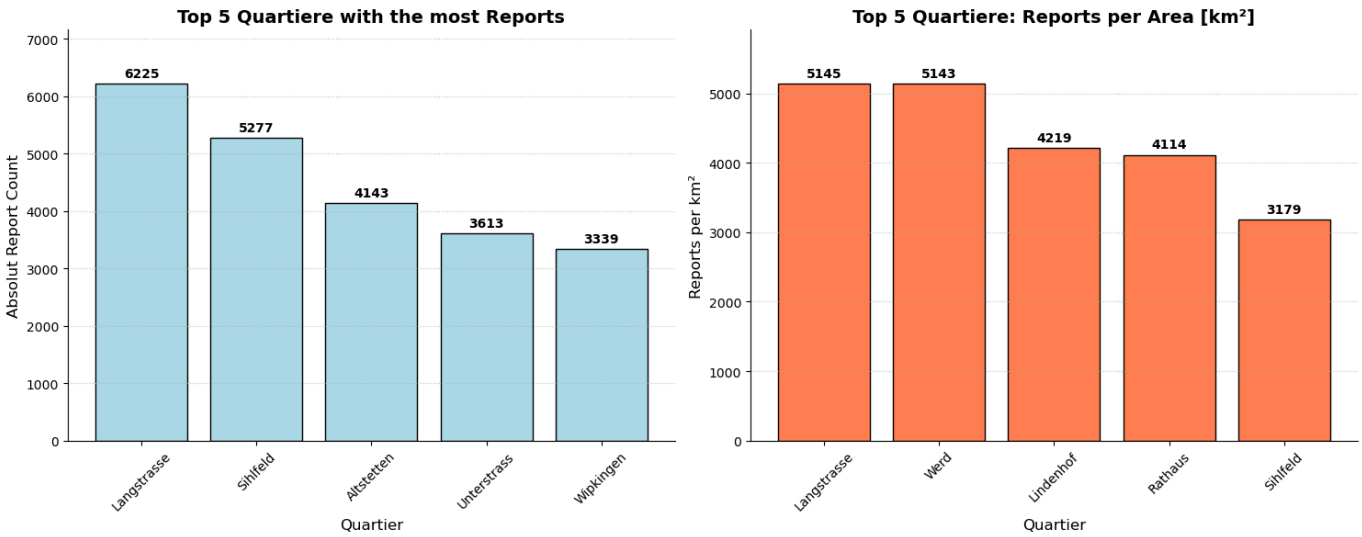
Question: Which *Quartiere* receive the highest number of reports?

- Report count per *Quartier*, normalized by the area.
- Reveals whether some *Quartiere* are genuinely more problematic, or just denser.
- Directly important and actionable for resource allocation.

In [6]:
```
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(16, 10))

final_quartiere_gdf.plot(
    column='report_count',
    scheme='fisherjenks',
    k=5,
    cmap='Blues',
    legend=True,
    ax=ax1,
    legend_kwds={'loc': 'lower right', 'title': 'Raw Report Count','fmt': '{:.0f}'},
    linewidth=0.3,
    edgecolor='black'
)
ax1.set_title('1. Raw Report Count per Quartier', fontsize=14, fontweight='bold')
ax1.set_axis_off()

final_quartiere_gdf.plot(
    column='reports_per_km2',
    scheme='fisherjenks',
    k=5,
    cmap='Reds',
    legend=True,
    ax=ax2,
    legend_kwds={'loc': 'lower right', 'title': 'Reports per km2', 'fmt': '{:.0f}'},
    linewidth=0.3,
    edgecolor='black'
)
ax2.set_title('2. Reports/km2 per Quartier', fontsize=14, fontweight='bold')
```

```
ax2.set_axis_off()
plt.savefig(PATH_OUTPUTS / 'figures' / 'q1_report_count_map.png', dpi=300, bbox_inche
plt.show()

top_5_count = final_quartiere_gdf.nlargest(5, 'report_count')
top_5_density = final_quartiere_gdf.nlargest(5, 'reports_per_km2')

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(15, 6))

bars1 = ax1.bar(top_5_count['qname'], top_5_count['report_count'], color='lightblue',
ax1.set_title('Top 5 Quartiere with the most Reports', fontsize=14, fontweight='semib
ax1.set_xlabel('Quartier', fontsize=12)
ax1.set_ylabel('Absolut Report Count', fontsize=12)
ax1.tick_params(axis='x', rotation=45)
ax1.grid(axis='y', linestyle='dotted', alpha=0.7)
ax1.bar_label(bars1, padding=3, fontweight='bold')
ax1.set_ylim(0, top_5_count['report_count'].max() * 1.15)
ax1.spines[['top', 'right']].set_visible(False)

bars2 = ax2.bar(top_5_density['qname'], top_5_density['reports_per_km2'], color='cora
ax2.set_title('Top 5 Quartiere: Reports per Area [km²]', fontsize=14, fontweight='bol
ax2.set_xlabel('Quartier', fontsize=12)
ax2.set_ylabel('Reports per km²', fontsize=12)
ax2.tick_params(axis='x', rotation=45)
ax2.grid(axis='y', linestyle='dotted', alpha=0.7)
ax2.bar_label(bars2, fmt='%.0f', padding=3, fontweight='bold')
ax2.set_ylim(0, top_5_density['reports_per_km2'].max() * 1.15)
ax2.spines[['top', 'right']].set_visible(False)

plt.tight_layout()
plt.savefig(PATH_OUTPUTS / 'figures' / 'q1_report_count_plot.png', dpi=300, bbox_inch
plt.show()
```





4.1.1 Answer

Absolute Counts:

- The top 5 Quartiers with the highest raw report counts are Langstrasse, Sihlfeld, Altstetten, Unterstrass, and Wipkingen. These top candidates are mostly found among the largest Quartiers in Zurich, regarding the area.

Report Density (per km2):

- When normalizing the raw counts by area, the spatial pattern changes entirely. The highest density of reports is found in Langstrasse, Werd, Lindenhof, Rathaus, and Sihlfeld. This shows that the core reporting activity is heavily concentrated in the built-up, high-traffic area in the city centre.

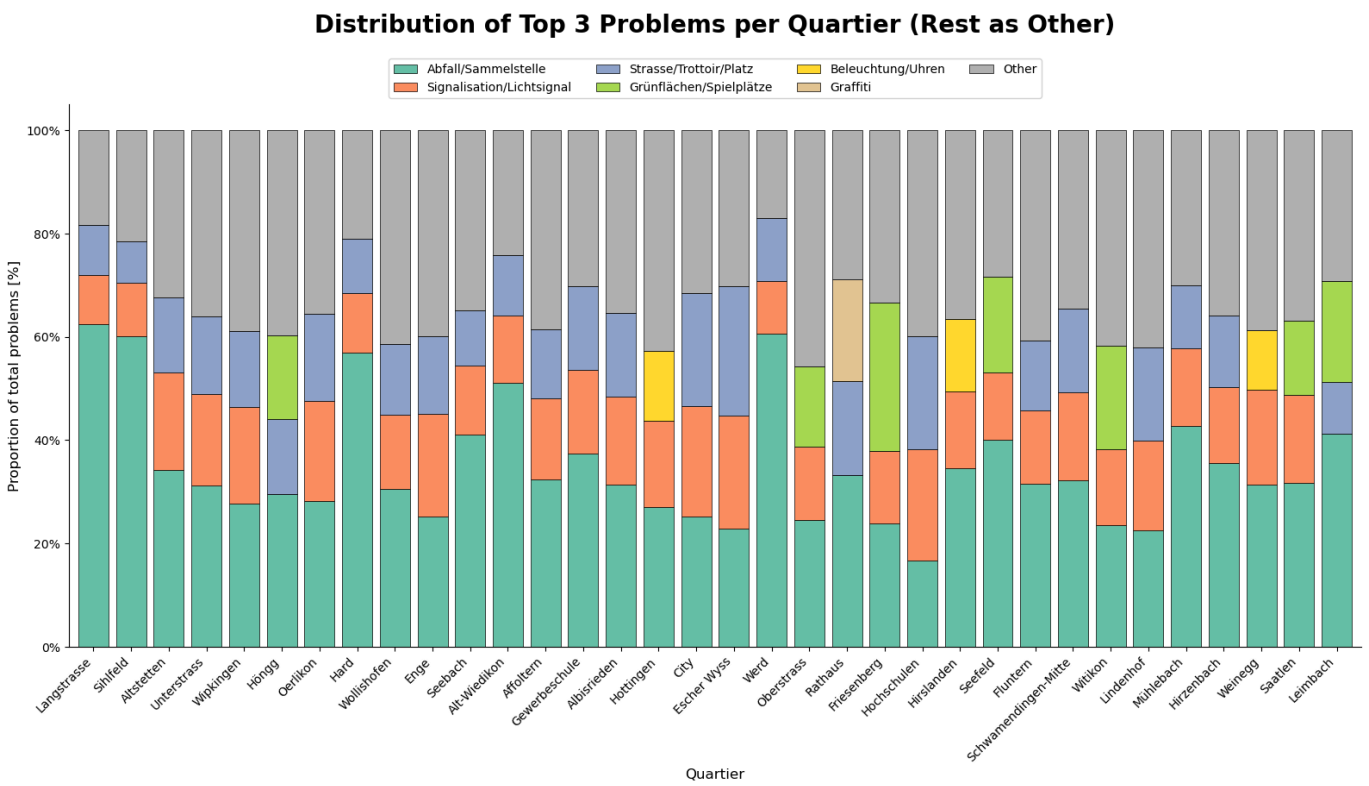
4.2 **Q2: Dominating Problems**

Question: Which problem-categories dominate in which Quartier?

- Relative share of Top 3 most common problems, together with the rest, summed up in *Other* per Quartier
- Map with most common Problem for each Quartier

In [7]:
```
cols_numeric   = category_count.select_dtypes(include=['number']).columns
df_numeric     = category_count[cols_numeric]
q_total_volume = df_numeric.sum(axis=1)

df_top3        = df_numeric.apply(lambda s: s.nlargest(3), axis=1)
df_top3['Other'] = q_total_volume - df_top3.sum(axis=1)

global_importance = (df_top3.drop(columns='Other').sum()
    .sort_values(ascending=False)
    .index.tolist() + ['Other'])

df_top3   = df_top3[global_importance]
df_final  = df_top3.div(q_total_volume, axis=0)
q_order   = q_total_volume.sort_values(ascending=False).index
df_final  = df_final.reindex(q_order)

fig, ax = plt.subplots(figsize=(16, 10))

df_final.plot(
```

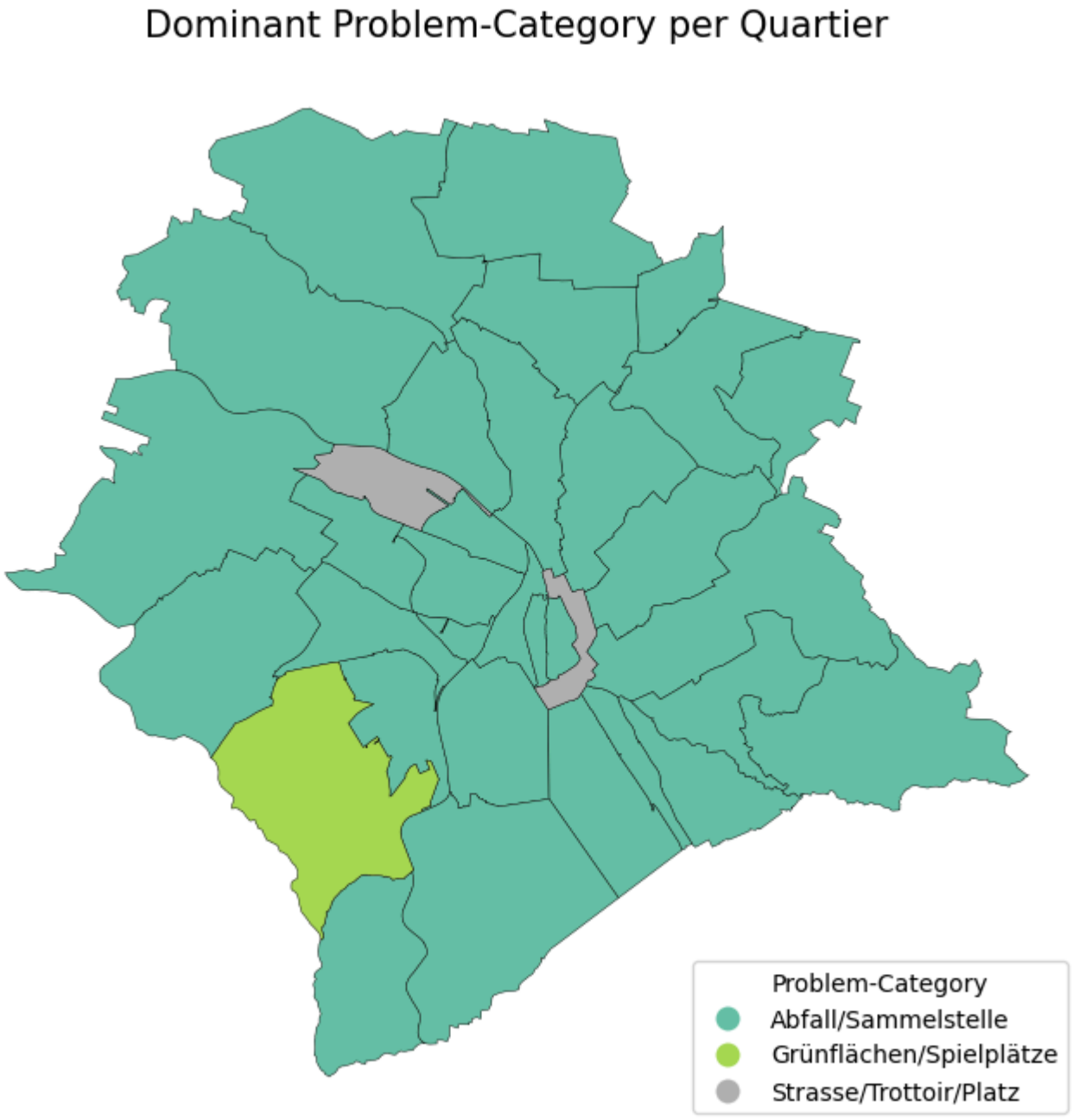
```
    kind='bar',
    stacked=True,
    ax=ax,
    colormap='Set2',
    width=0.8,
    edgecolor='black',
    linewidth=0.5
)
plt.title('Distribution of Top 3 Problems per Quartier (Rest as Other)', fontsize=20,
plt.xlabel('Quartier', fontsize=12)
plt.ylabel('Proportion of total problems [%]', fontsize=12)
ax.yaxis.set_major_formatter(mtick.PercentFormatter(1.0))
plt.xticks(rotation=45, ha='right')
ax.spines[['top', 'right']].set_visible(False)
plt.legend(
    loc='lower center',
    bbox_to_anchor=(0.5, 1.0),
    ncol=4,
    fontsize=10,
    frameon=True
)
plt.tight_layout()
plt.subplots_adjust(top=0.8)
plt.savefig(PATH_OUTPUTS / 'figures' / 'q2_top3_problems_distribution.png', dpi=300,
plt.show()
```



In [8]:

```
ax = final_quartiere_gdf.plot(
    figsize=(10, 8),
    column='top_category',
    categorical=True,
    legend=True,
    legend_kwds={'loc': 'lower right',
        'title': 'Problem-Category'},
    cmap='Set2',
    edgecolor='black',
    linewidth=0.3
)
ax.set_title('Dominant Problem-Category per Quartier', fontsize=15, pad=10)
ax.set_axis_off()
```

```
plt.savefig(PATH_OUTPUTS / 'figures' / 'q2_top_problem_map.png', dpi=300, bbox_inches
plt.show()
```



4.2.1 Answer

City-wide Baseline:

- The Top 3 problem-categories across the entire city are mostly the same: Abfall/Sammelstelle (Waste/Collection points), Strasse/Trottoir/Platz (Streets/Sidewalks/Squares), and Signalisation/Lichtsignal (Signage/Traffic Lights).

Dominance & Exceptions:

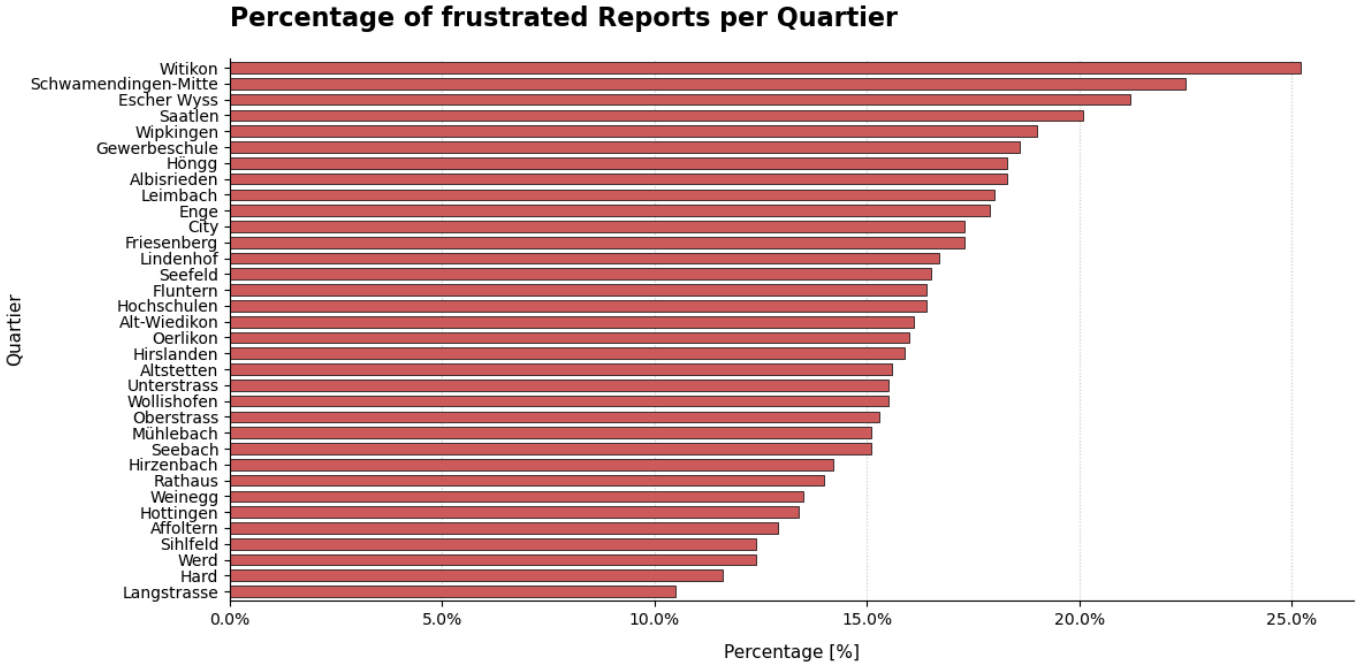
- The category Abfall/Sammelstelle is the number one issue in almost every neighborhood. There are only three exceptions across the city: In Escher Wyss and City, the dominant category is Strasse/Trottoir/Platz, while Friesenberg is dominated by Grünflächen/Spielplätze (Green spaces/Playgrounds).

4.3 **Q3: Frustrated Reports**

Question: What is the proportion of frustrated Reports in each Quartier, and how are the frustrated Reports clustered over the city?

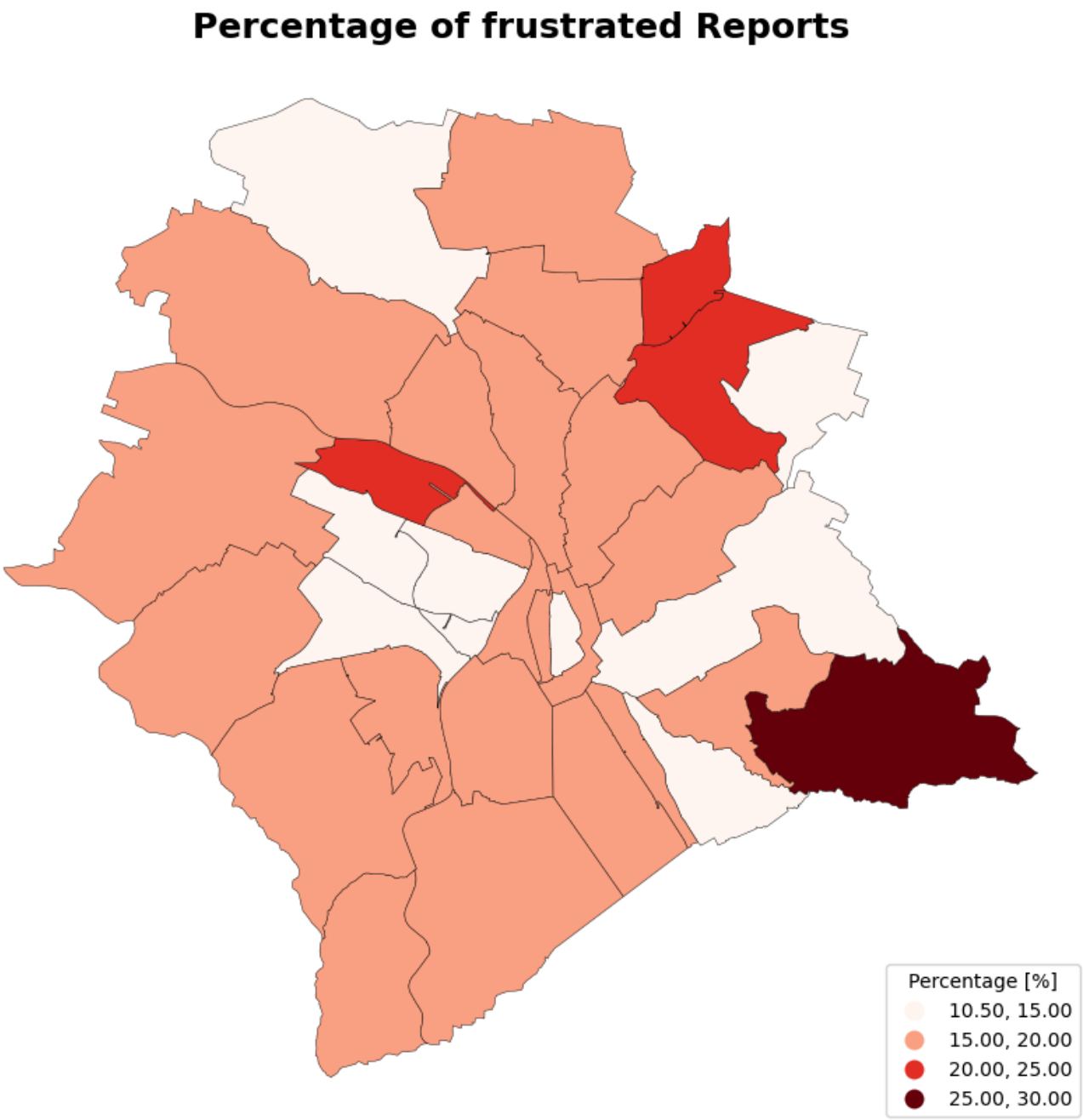
- Percentage of frustrated reports in each Quartier (Graph & Map)
- Clusters of frustrated reports over the whole city (DBSCAN)

In [9]:
```
fig, ax = plt.subplots(figsize=(12, 6), dpi=100)

(final_quartiere_gdf
 .sort_values('frustration_rate', ascending=True)
 .set_index('qname')
 ['frustration_rate']
 .plot(
     kind='barh',
     color='indianred',
     width=0.7,
     ax=ax,
     edgecolor='black',
     linewidth=0.5
 )
)
ax.set_title('Percentage of frustrated Reports per Quartier', fontsize=16, fontweight
ax.set_xlabel('Percentage [%]', fontsize=11, labelpad=10)
ax.set_ylabel('Quartier', fontsize=11)
ax.xaxis.set_major_formatter(mtick.PercentFormatter(100.0))
ax.xaxis.grid(True, linestyle='dotted', alpha=0.6)
ax.set_axisbelow(True)
ax.spines[['top', 'right']].set_visible(False)
plt.tight_layout()
plt.savefig(PATH_OUTPUTS / 'figures' / 'q3_frustration_plot.png', dpi=300, bbox_inche
plt.show()
```



In [10]:
```
ax = final_quartiere_gdf.plot(
    figsize=(12, 10),
    column='frustration_rate',
    scheme='prettybreaks',
    cmap='Reds',
    legend=True,
    legend_kwds={'loc': 'lower right',
        'title': 'Percentage [%]'},
    linewidth=0.3,
```

```
    edgecolor='black',
)
ax.set_title('Percentage of frustrated Reports', fontsize=18, fontweight='bold')
ax.set_axis_off()
plt.savefig(PATH_OUTPUTS / 'figures' / 'q3_frustration_map.png', dpi=300, bbox_inches
plt.show()
```

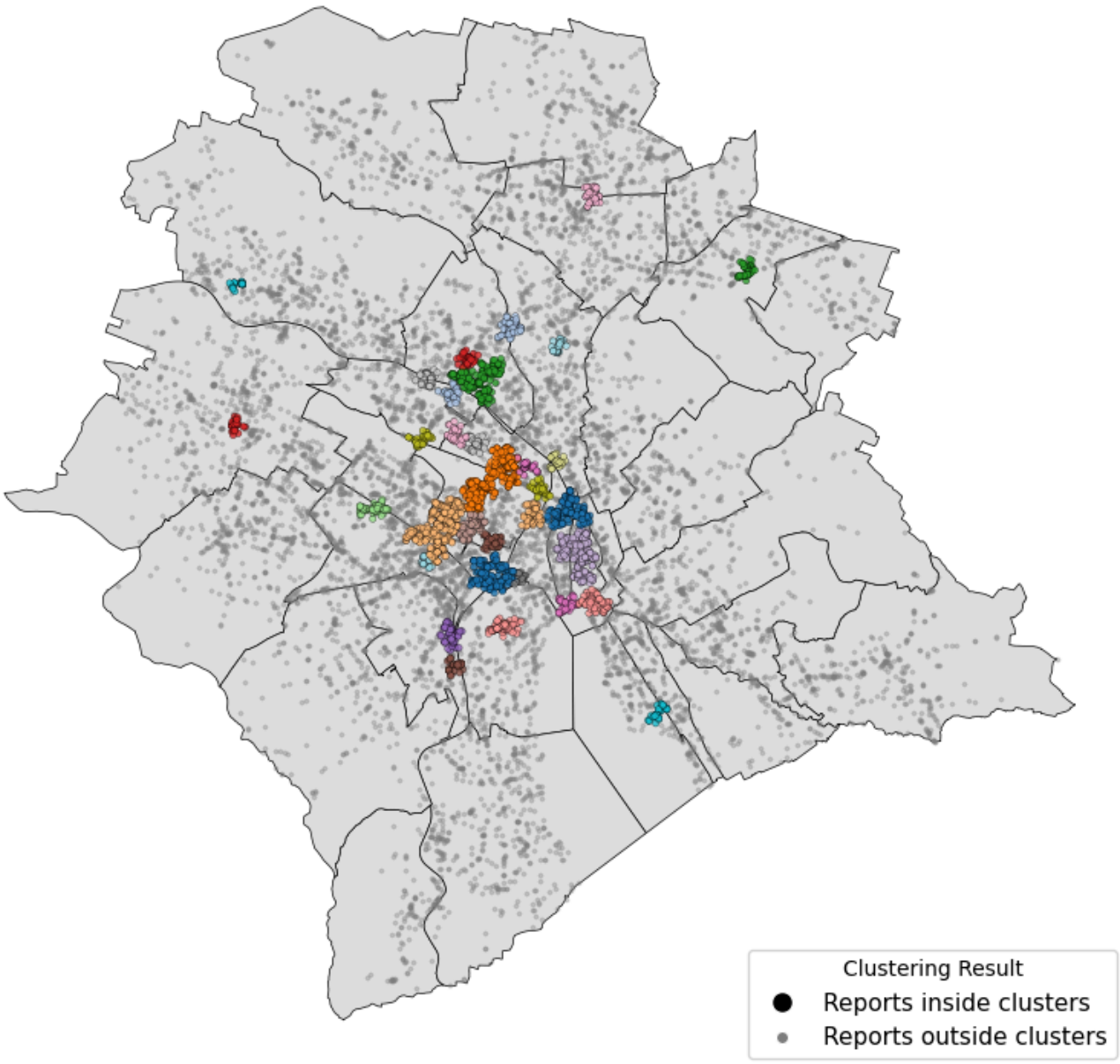


In [11]:

```
eps = 100
min_samples = 30

frustrated_gdf, dbscan_clusters, dbscan_noise, cluster_polygons = ut.reports_clusteri

fig, ax = plt.subplots(1, 1, figsize=(10, 8))

final_quartiere_gdf.plot(
    ax=ax,
    color='#e0e0e0',
    edgecolor='black',
    linewidth=0.5
)
dbscan_noise.plot(
    ax=ax,
    color='grey',
```

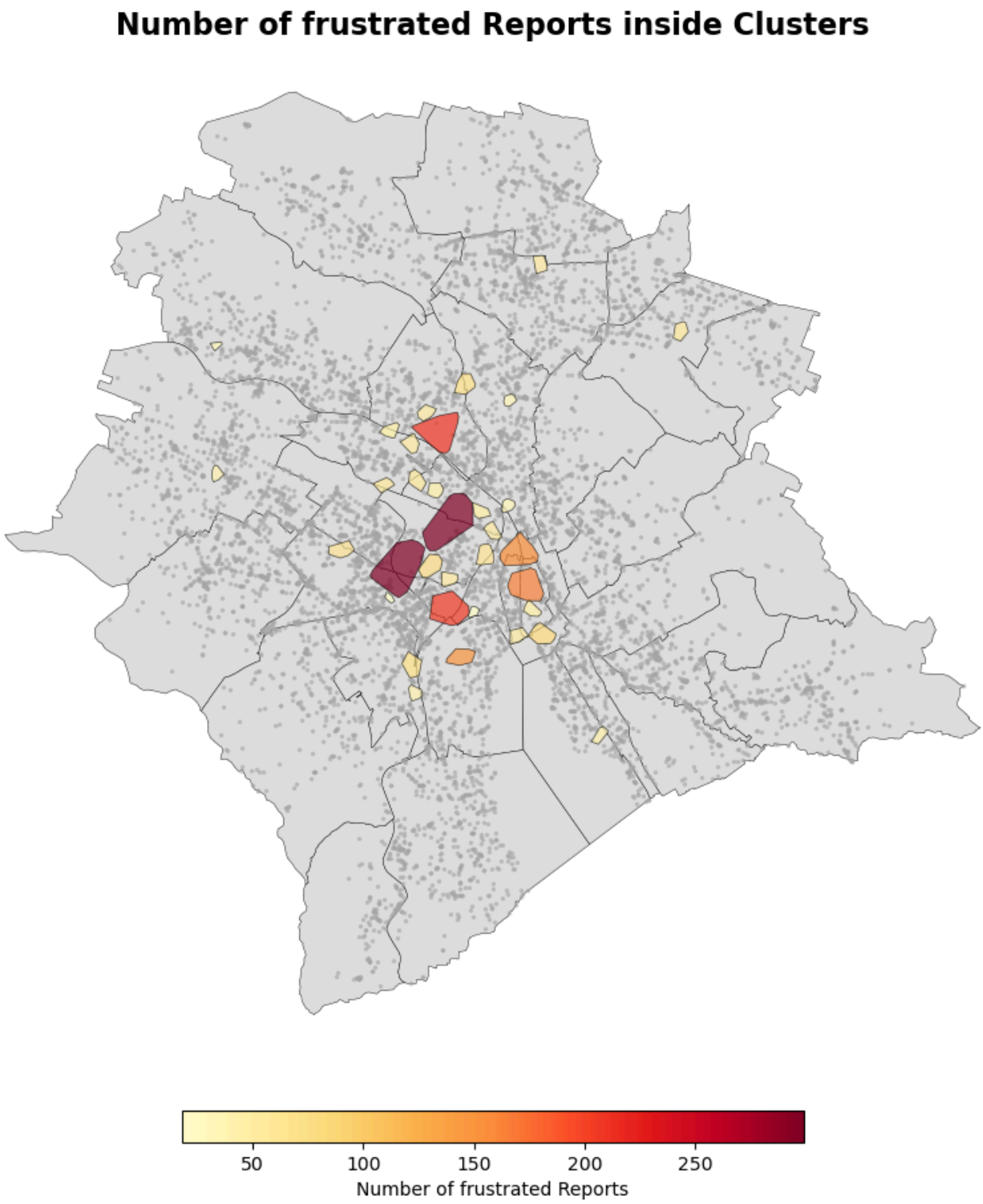
```
    markersize=3,
    alpha=0.3,
    label='Reports outside clusters'
)
dbscan_clusters.plot(
    column='cluster_id',
    ax=ax,
    cmap='tab20',
    markersize=10,
    alpha=0.8,
    edgecolor='black',
    linewidth=0.25
)
ax.set_title('Frustration-Hotspots: Density-Based Clustering (DBSCAN)', fontsize=16,
ax.set_axis_off()

from matplotlib.lines import Line2D
legend_elements = [
    Line2D([0], [0], marker='o', color='w', label='Reports inside clusters', markerfa
    Line2D([0], [0], marker='o', color='w', label='Reports outside clusters', markerf
]
ax.legend(handles=legend_elements, loc='lower right', title='Clustering Result', font
plt.tight_layout()

num_clusters = len(cluster_polygons)
print(f'DBSCAN has idenified {num_clusters} frustration-Hotspots')
print(f'Out of {len(frustrated_gdf)} frustrated Reports, {len(dbscan_clusters)} belon

plt.savefig(PATH_OUTPUTS / 'figures' / 'q3_dbscan_points_map.png', dpi=300, bbox_inch
plt.show()
```

```
DBSCAN has idenified 34 frustration-Hotspots
Out of 11435 frustrated Reports, 2822 belong to a Cluster.
```



In [12]:
```
fig, ax = plt.subplots(1, 1, figsize=(12, 10))

final_quartiere_gdf.plot(
    ax=ax,
    color='#e0e0e0',
    edgecolor='black',
    linewidth=0.3
)
dbscan_noise.plot(
    ax=ax,
    color='darkgrey',
    markersize=2.5,
    alpha=0.5,
    label='Noise'
)
cluster_polygons.plot(
    ax=ax,
    column='n_points',
    cmap='YlOrRd',
    alpha=0.7,
    edgecolor='black',
    linewidth=0.5,
```

```
    legend=True,
    legend_kwds={
        'label': 'Number of frustrated Reports',
        'orientation': 'horizontal',
        'pad': 0.04,
        'shrink': 0.4
    }
)
ax.set_title('Number of frustrated Reports inside Clusters', fontsize=16, fontweight=
ax.set_axis_off()
plt.tight_layout()
plt.savefig(PATH_OUTPUTS / 'figures' / 'q3_dbscan_cluster_map.png', dpi=300, bbox_inc
plt.show()
```



4.3.1 Answer

Proportional Rate on Quartier-level:

- The peripheral and residential Quartiers of Witikon, Schwamendingen-Mitte, Escher Wyss, and Saatlen stand out with the highest relative proportion of frustrated complaints, all exceeding 20%. Aggregating this rate to the Quartiers shows no distinct concentration towards the center.

Spatial Clustering (DBSCAN):

- In contrast, the density-based DBSCAN clustering reveals that the dense, physical hotspots of frustrated reports are heavily concentrated in the city center. This is a classic spatial scale effect: because the center has a higher volume of overall traffic and reporting, the mathematical probability of absolute clusters forming there is much higher, even if peripheral districts have a higher relative percentage of frustrated reports.

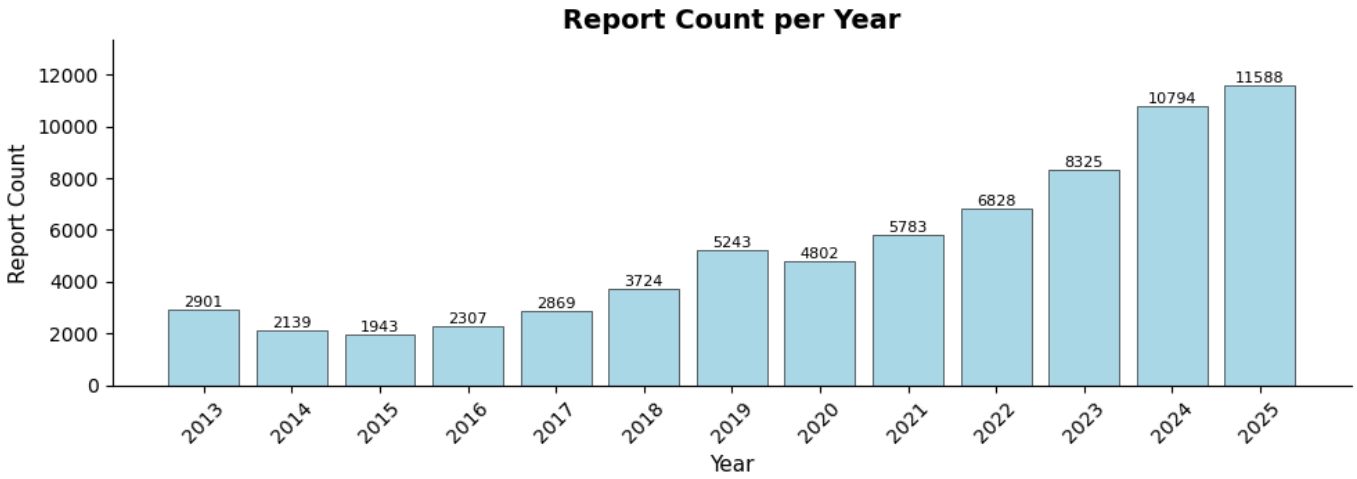
4.4 **Q4 Temporal Trend and Rate of Change**

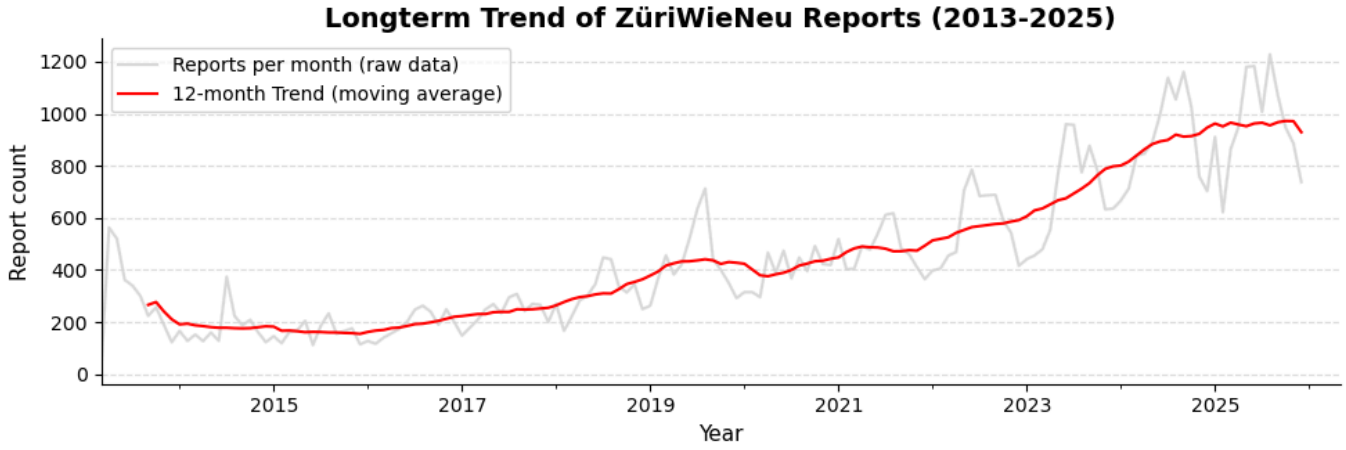
Question: How has the amount of Reports changed over time, and in which Quartiere is the Rate of Change significantly higher?

- Compare average of two periods (e.g. 2013-2018 to 2019-2025)
- Logic: (New_Period - Old_Period) / Old_Period = Rate of Change

In [13]:
```
annual = (
    final_reports_gdf
    .groupby('year')
    .size()
    .reset_index(name='count')
)
annual = annual[annual['year'] < 2026]

monthly_reports = (
    final_reports_gdf
    .set_index('requested_datetime')
    .resample('ME')
    .size()
)
trend_12m = monthly_reports.rolling(window=12, center=True).mean()
monthly_reports = monthly_reports.loc[:'2025']

fig, (ax1, ax2) = plt.subplots(nrows=2, ncols=1, figsize=(10, 7))

bars = ax1.bar(annual['year'], annual['count'], color='lightblue', edgecolor='black',
ax1.bar_label(bars, fontsize=8)
ax1.set_xlabel('Year', fontsize=11)
ax1.set_ylabel('Report Count', fontsize=11)
ax1.set_title('Report Count per Year', fontsize=14, fontweight='bold')
ax1.set_xticks(annual['year'])
ax1.tick_params(axis='x', rotation=45)
ax1.spines[['top', 'right']].set_visible(False)
ax1.set_ylim(0, annual['count'].max() * 1.15)

monthly_reports.plot(ax=ax2, color='lightgrey', alpha=0.8, label='Reports per month (
trend_12m.plot(ax=ax2, color='red', linewidth=1.5, label='12-month Trend (moving aver
ax2.set_title('Longterm Trend of ZüriWieNeu Reports (2013-2025)', fontsize=14, fontwe
ax2.set_ylabel('Report count', fontsize=11)
ax2.set_xlabel('Year', fontsize=11)
ax2.grid(axis='y', linestyle='--', alpha=0.4)
ax2.spines[['top', 'right']].set_visible(False)
ax2.legend()

plt.tight_layout()
```

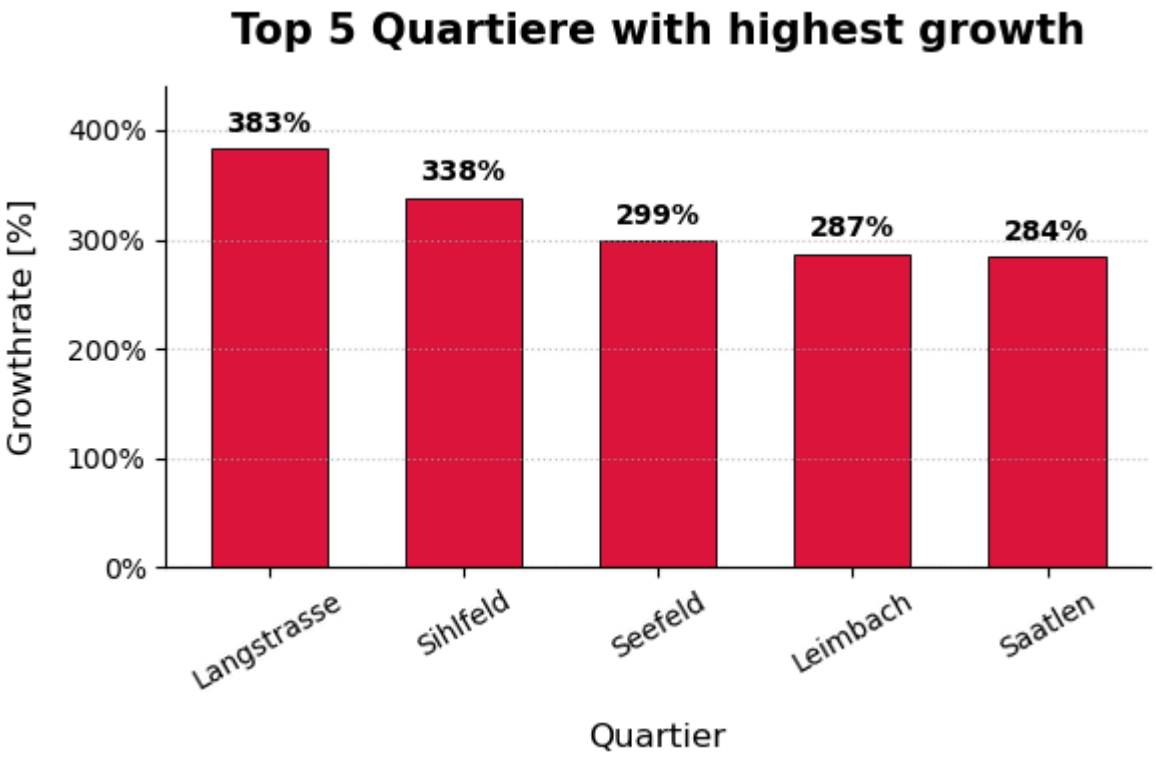
```
plt.savefig(PATH_OUTPUTS / 'figures' / 'q4_trends_plot.png', dpi=300, bbox_inches='ti
plt.show()
```

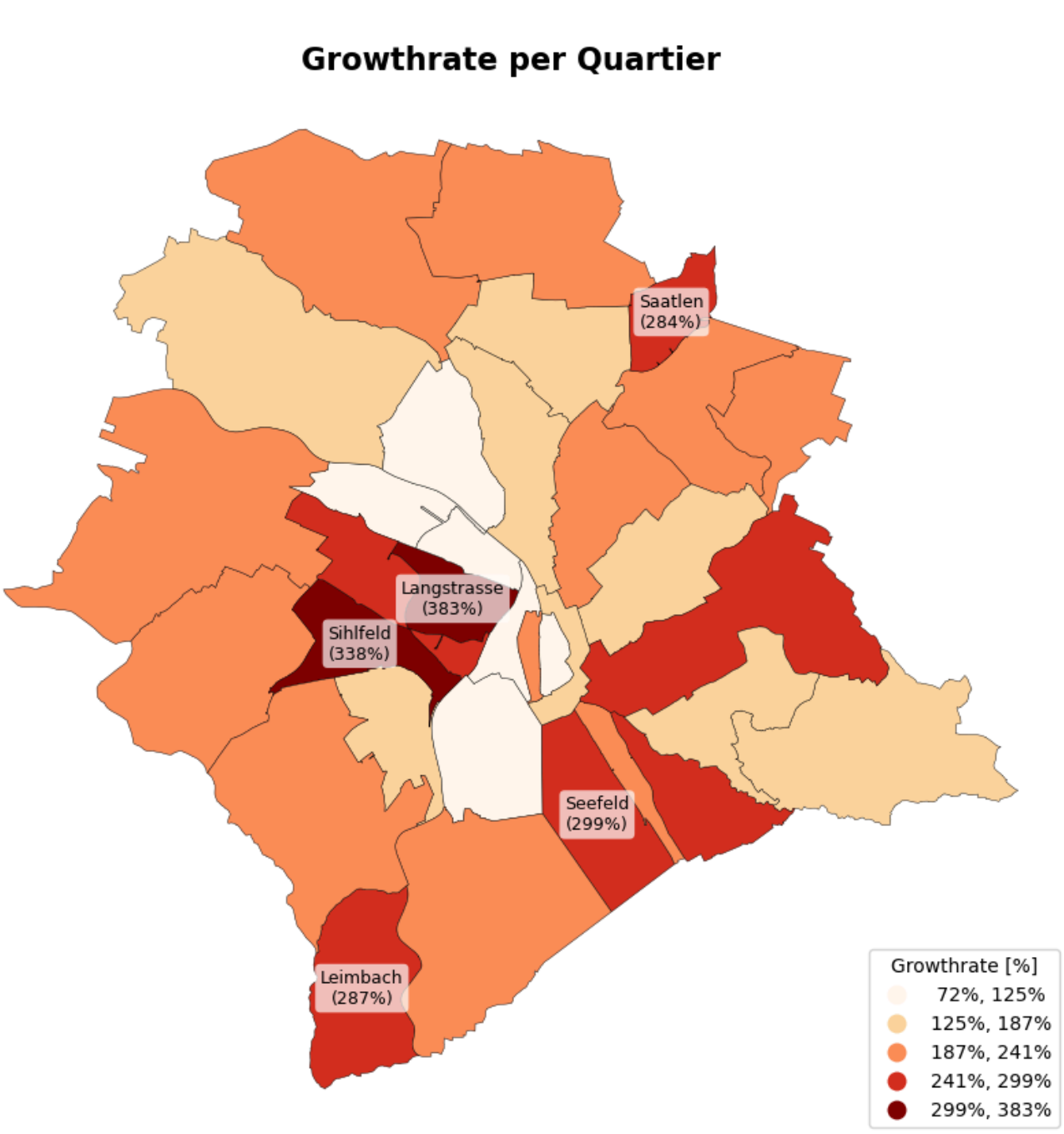




In [14]:
```
top5_growth = final_quartiere_gdf.nlargest(5, 'growth_rate')

fig, ax = plt.subplots(figsize=(6, 4))

bars = ax.bar(
    top5_growth['qname'],
    top5_growth['growth_rate'],
    color='crimson',
    edgecolor='black',
    linewidth=0.6,
    width=0.6
)
ax.set_title('Top 5 Quartiere with highest growth', fontsize=15, fontweight='bold', p
ax.set_xlabel('Quartier', fontsize=12, labelpad=10)
ax.set_ylabel('Growthrate [%]', fontsize=12, labelpad=10)
ax.bar_label(bars, fmt='{:.0f}%', padding=4, fontweight='bold', fontsize=10)
ax.yaxis.set_major_formatter(mtick.PercentFormatter(100.0))
ax.tick_params(axis='x', rotation=30)
ax.grid(axis='y', linestyle='dotted', alpha=0.7)
ax.spines[['top', 'right']].set_visible(False)
ax.set_ylim(0, top5_growth['growth_rate'].max() * 1.15)
plt.tight_layout()
plt.savefig(PATH_OUTPUTS / 'figures' / 'q4_top5_growth_plot.png', dpi=300, bbox_inche
plt.show()

fig, ax = plt.subplots(figsize=(12, 10))

final_quartiere_gdf.dropna(subset=['growth_rate']).plot(
    ax=ax,
    column='growth_rate',
    scheme='fisherjenks',
    k=5,
```

```
    cmap=cmap_to_use,
    edgecolor='black',
    linewidth=0.3,
    legend=True,
    legend_kwds={
        'title': 'Growthrate [%]',
        'loc': 'lower right',
        'fmt': '{:.0f}%'
    }
)

top5 = final_quartiere_gdf.nlargest(5, 'growth_rate')
top5.apply(lambda row: ax.annotate(
    text=f"{row['qname']}\n({row['growth_rate']:.0f}%)",
    xy=(row.geometry.centroid.x, row.geometry.centroid.y),
    fontsize=9,
    ha='center',
    va='center',
    bbox=dict(facecolor='white', alpha=0.7, edgecolor='none', boxstyle='round,pad=0.3
), axis=1)
ax.set_title('Growthrate per Quartier', fontsize=16, fontweight='bold')
ax.set_axis_off()
plt.savefig(PATH_OUTPUTS / 'figures' / 'q4_growth_map.png', dpi=300, bbox_inches='tig
plt.show()
```





4.4.1 Answer

Temporal Trend:

- Prior to 2019 (2013 – 2018), the annual number of reported issues remained relatively stable. After 2019 (2020–2025), platform usage increased significantly, with reporting numbers more than doubling. The year 2020 experienced a temporary drop compared to 2019, which is likely a direct footprint of restricted public mobility during the COVID-19 pandemic. But this is just a speculation. More context would be needed to clarify it.

Growth Rates:

- When comparing the pre- and post-2019 periods, Langstrasse clearly leads the city with an huge growthrate of 383%. It is followed by Sihlfeld (338%) and Seefeld (299%). High-growth Quartiers are distributed across various parts of Zurich and are not only concentrated to the city center.

4.5 Saving processed Data

In [15]:
```
final_quartiere_gdf.to_file(PATH_PROCESSED_DATA / 'quartiere_metrics.gpkg', driver='G
final_reports_gdf.to_file(PATH_PROCESSED_DATA / 'reports_metrics.gpkg', driver='GPKG'
cluster_polygons.reset_index().to_file(PATH_PROCESSED_DATA / 'frustration_clusters.gp

growth_df.to_csv(PATH_OUTPUTS / 'data'  / 'growth_rates.csv', index=True)
category_count.to_csv(PATH_OUTPUTS / 'data'  / 'category_by_quartier.csv')

print('All Outputs saved!')
print('GPKG:', PATH_PROCESSED_DATA)
print('CSV:', PATH_OUTPUTS / 'data')
```

```
All Outputs saved!
GPKG: /Users/maxlengenfelder/Desktop/SDS210_ZuriWieNeu/data/processed
CSV: /Users/maxlengenfelder/Desktop/SDS210_ZuriWieNeu/outputs/data
```

5. Summary

This spatial and temporal analysis demonstrates that public problem-reporting via the ZüriWieNeu platform follows structured geographic patterns rather than a random distribution. While raw reporting metrics are easily skewed by a Quartier's area, using area-normalization and spatial clustering (DBSCAN) exposes the city center and major commercial and build-up zones as the primary area of infrastructure problems.

Methodologically, the divergence in these findings shows the importance of selecting the appropriate spatial unit of analysis (areal aggregation versus point clustering for example) to avoid misleading and wrong assumptions.

From an administrative perspective, the findings provide a solid, data-driven foundation for optimizing public maintenance schedules and resource allocation. However, an analytical reflection must be made: a high concentration of crowd-sourced reports does not strictly imply the decaying of physical infrastructure. Reporting activity is highly linked with underlying socio-demographic variables, population density, and fluctuating periods of citizen engagement. Future improvements of this analysis should integrate these external factors to move from descriptive spatial patterns to true causal urban insights, adding the necessary context.

5.1 Summary of Findings and Conclusion

In [16]:
```
ut.summary_questions(final_reports_gdf, final_quartiere_gdf, cluster_polygons)
```

```
======================================================
RESULTS: ZüriWieNeu-Analysis                  2013–2025
======================================================
Analysed Reports:                                73,337
Time period:                                2013 – 2025
Quartiere covered:                                   34

Q1 – Highest Report density:                  Langstrasse
                                    (5145 Reports/km2)

Q2 – Dominant Problem-Category:     Abfall/Sammelstelle

Q3 – Highest frustration-Rate:             Witikon (25.2%)
     DBSCAN:          34 Frustration-Clusters identified

Q4 – Strongest Growth:                Langstrasse (+383%)
======================================================
```

5.2 Interactive Map

In [17]:

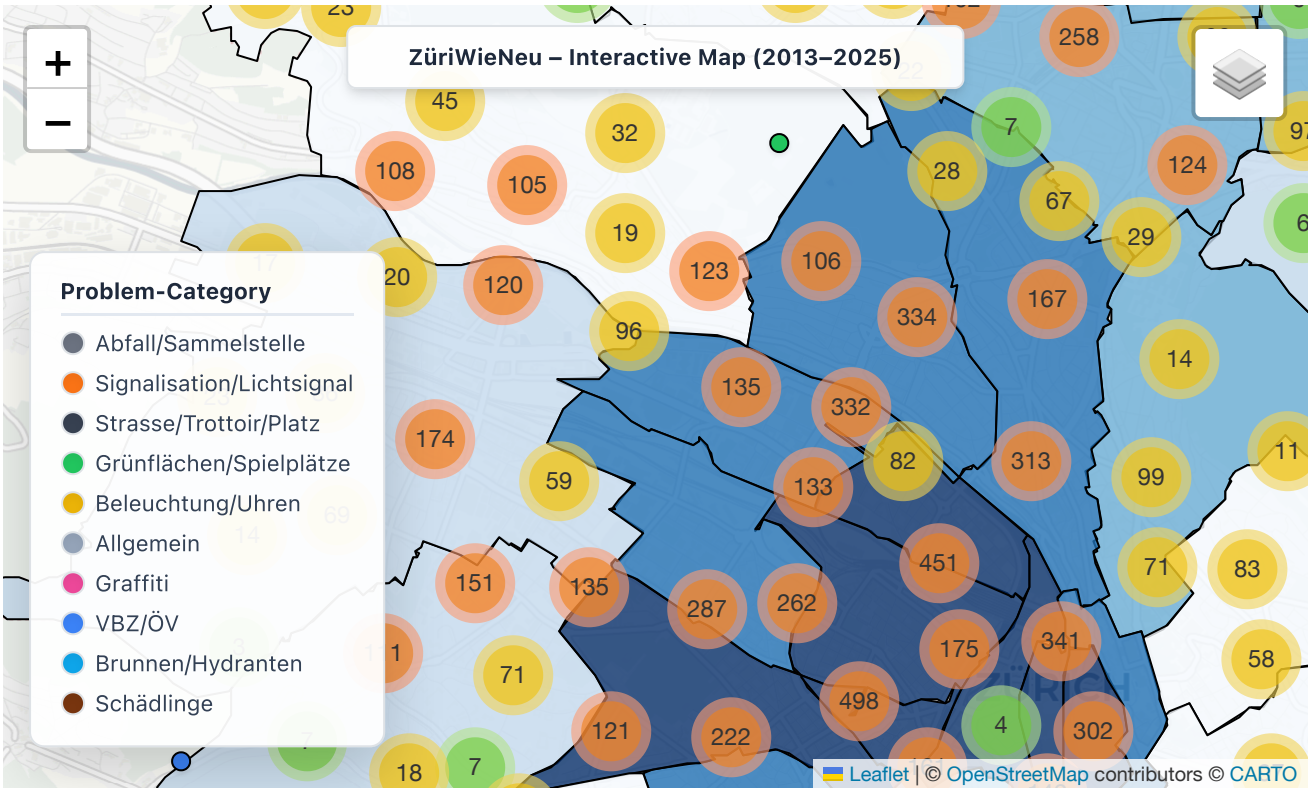
```
CAT_COLORS = {
    'Abfall/Sammelstelle':          '#6b7280',
    'Signalisation/Lichtsignal':    '#f97316',
    'Strasse/Trottoir/Platz':       '#374151',
    'Grünflächen/Spielplätze':      '#22c55e',
    'Beleuchtung/Uhren':            '#eab308',
    'Allgemein':                    '#94a3b8',
    'Graffiti':                     '#ec4899',
    'VBZ/ÖV':                       '#3b82f6',
    'Brunnen/Hydranten':            '#0ea5e9',
    'Schädlinge':                   '#78350f',
}

quartiere_4326   = final_quartiere_gdf.to_crs(epsg=4326)
reports_4326     = final_reports_gdf.to_crs(epsg=4326)
cluster_4326     = cluster_polygons.reset_index().to_crs(epsg=4326)
frustrated_4326  = frustrated_gdf.to_crs(epsg=4326)

m = folium.Map(location=[47.3769, 8.5417], zoom_start=13, tiles=None, max_zoom=19)

m.get_root().html.add_child(folium.Element('<script src="https://cdn.tailwindcss.com"

folium.TileLayer('CartoDB Positron', name='Basemap: CartoDB (Dark)', show=True).add_t
folium.TileLayer(
    tiles='https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServ
    attr='Esri World Imagery', name='Basemap: Sateliteimage', show=False
).add_to(m)

STYLE = {'fillOpacity': 0.8, 'color': 'black', 'weight': 1.0}

quartiere_4326.explore(
    m=m, column='reports_per_km2', cmap='Blues', scheme='quantiles', k=5,
    name='Q1: Report density per km2', show=True, legend=False,
    tooltip=['qname', 'reports_per_km2', 'report_count'],
    tooltip_kwds={'aliases': ['Quartier', 'Reports/km2', 'Reports total'], 'localize'
    style_kwds=STYLE
)

quartiere_4326.explore(
    m=m, column='top_category', cmap='Set2',
    name='Q2: Dominant Problem Category', show=False, legend=False,
    tooltip=['qname', 'top_category'],
    tooltip_kwds={'aliases': ['Quartier', 'Dominant Category']},
    style_kwds=STYLE
)

quartiere_4326.explore(
    m=m, column='frustration_rate', cmap='OrRd', scheme='quantiles', k=5,
    name='Q3a: Frustration rate per Quartier', show=False, legend=False,
    tooltip=['qname', 'frustration_rate', 'report_count'],
    tooltip_kwds={'aliases': ['Quartier', 'Frustration Rate [%]', 'Reports total'], '
    style_kwds=STYLE
)

marker_cluster = MarkerCluster(
    name='Q3b: Frustrated Reports (Cluster)',
    options={'maxClusterRadius': 40, 'disableClusteringAtZoom': 17, 'spiderfyOnMaxZoo
).add_to(m)

for _, row in frustrated_4326.iterrows():
```

```
    cat = row['service_name']
    color = CAT_COLORS.get(cat, '#888888')

    full_detail = str(row.get('detail', 'No Details available'))
    short_detail = full_detail[:70] + '...' if len(full_detail) > 70 else full_detail

    popup_html = f"""
    <div class='font-sans text-lg min-w-[220px] max-w-[280px]'>
        <h4 class='font-bold text-slate-800 border-b border-slate-200 pb-1 mb-2'>{cat
        <p class='text-slate-600 whitespace-pre-wrap max-h-48 overflow-y-auto'>{full_
    </div>
    """

    folium.Marker(
        location=[row.geometry.y, row.geometry.x],
        icon=folium.DivIcon(
            html=f"<div style='background:{color};' class='w-4 h-4 rounded-full borde
            icon_size=(16, 16),
            icon_anchor=(8, 8),
        ),
        tooltip=f"<span class='font-sans'><b class='text-slate-800'>{cat}</b><br><i c
        popup=folium.Popup(popup_html, max_width=300)
    ).add_to(marker_cluster)

cluster_4326.explore(
    m=m, column='n_points', cmap='YlOrRd',
    name='Q3c: DBSCAN Cluster', show=False, legend=False,
    tooltip=['cluster_id', 'n_points'],
    tooltip_kwds={'aliases': ['Cluster ID', 'Frustrated Reports']},
    style_kwds={'fillOpacity': 0.6, 'color': 'black', 'weight': 1.0}
)

quartiere_4326.explore(
    m=m, column='growth_rate', cmap=cmap_to_use,
    name='Q4: Growthrate (2013–2025)', show=False, legend=False,
    tooltip=['qname', 'growth_rate'],
    tooltip_kwds={'aliases': ['Quartier', 'Growthrate [%]'], 'localize': True},
    style_kwds=STYLE
)

folium.LayerControl(collapsed=True, position='topright').add_to(m)

legend_items = "".join([
    f"<div class='flex items-center gap-2 mb-1'><span class='w-5 h-5 rounded-full inl
    for cat, c in CAT_COLORS.items()
])

legend_html = f"""
<div id="category-legend" class="hidden fixed bottom-8 left-5 z-[1000] bg-white/95 ba
    <div class="font-bold text-slate-800 mb-2 border-b border-slate-200 pb-1">Problem
    {legend_items}
</div>
"""
m.get_root().html.add_child(folium.Element(legend_html))

title_html = '''
<div class='fixed top-4 left-1/2 -translate-x-1/2 z-[1000] bg-white/95 backdrop-blur-
    ZüriWieNeu – Interactive Map (2013–2025)
</div>
'''
m.get_root().html.add_child(folium.Element(title_html))

interactive_js = '''
```

```
<script>
document.addEventListener('DOMContentLoaded', function() {
    setTimeout(function() {
        const inputs = document.querySelectorAll('.leaflet-control-layers-overlays in
        const legend = document.getElementById('category-legend');

        function updateUI() {
            let showLegend = false;
            document.querySelectorAll('.leaflet-control-layers-overlays label').forEa
                const input = label.querySelector('input');
                if (input && input.checked) {
                    const text = label.textContent;
                    // Show legend ONLY if Q2 or Q3b is active
                    if (text.includes('Q2') || text.includes('Q3b')) {
                        showLegend = true;
                    }
                }
            });
            if (legend) {
                legend.classList.toggle('hidden', !showLegend);
            }
        }

        inputs.forEach(cb => {
            cb.addEventListener('change', function() {
                if (this.checked) {
                    inputs.forEach(other => {
                        if (other !== this && other.checked) { other.click(); }
                    });
                }
                // Update legend after Leaflet processes the change
                setTimeout(updateUI, 50);
            });
        });

        // Initial check on load
        updateUI();
    }, 600);
});
</script>
'''
m.get_root().html.add_child(folium.Element(interactive_js))

m.save(str(PATH_OUTPUTS / 'figures' / 'interactive_map.html'))
print('Interactive Map saved!')
m
```

```
Interactive Map saved!
```

Out[17]:


ZüriWieNeu – Interactive Map (2013–2025)
Problem-Category
Abfall/Sammelstelle
Signalisation/Lichtsignal
Strasse/Trottoir/Platz
Grünflächen/Spielplätze
Beleuchtung/Uhren
Allgemein
Graffiti
VBZ/ÖV
Brunnen/Hydranten
Schädlinge
Leaflet | © OpenStreetMap contributors © CARTO