Data Visualization Assignment 3: Group 16

Oriana Di Nisi, Enrico Saro, Zeinab Sepehrirezaeian

In this document we describe and explain the dashboard that was built for the Data Visualization for and with AI assignment. It explores Airbnb listings in Ghent using data from Inside Airbnb (https://insideairbnb.com/get-the-data/). It shows where listings are located, how prices are distributed, and how supply is divided by room type. The goal is to let a user filter the dataset, focus on specific listings on the map, and immediately see how the distribution changes.

Design. The interface consists of three parts presented in a single page: filters at the top, a central interactive map of listings, and two summary charts below (price distribution and room-type composition). This arrangement supports a simple workflow: set a subset with the filters, explore it spatially on the map, then read the summaries that quantify the current subset.

Interaction. The map is the main entry point. Hovering the mouse over a *point* reveals detailed information for that specific listing (position, host name, price, room type). The built-in toolbar in the top-right of the map provides standard controls: *pan* to drag the viewport; *zoom in/out* to change scale and *select* tools including box and lasso. Using box or lasso creates an explicit selection of visible listings; the two charts then update to reflect the selected subset, not just the filtered set. A *clear selection* button placed next to the map instructions resets the selection, returning the view to the current filters' baseline (also double-click resets the selection).

Filtering. Filters include a multi-select for room type, a price range slider, and a minimum-nights slider. The map immediately reflects these predicates by showing only matching points. The charts always mirror the active state: when only filters are set, they summarize the filtered dataset; when a selection on the map is active, they summarize the selected listings.

Implementation. The system is built with Dash and Plotly; Bootstrap components handle layout. We split the code in four scripts. *layout.py* declares the page structure and stable component identifiers (filters, map, charts). *callbacks.py* defines two callbacks: a small reset callback to clear the map selection, and a main callback that loads the data, applies filter predicates, handles hover/selection events, and returns the three figures so that map and charts stay synchronized. *charts.py* provides data access and figure construction: it loads and cleans the CSV once (lightweight caching), and builds the Mapbox scatter layer, including a stable listing identifier when available so selections are matched reliably. *app.py* wires everything: it creates the Dash app, attaches the layout, and registers the callbacks. This separation keeps responsibilities clear: the map interactions update the current selection, the callbacks compute the corresponding summaries and redraw the charts, while data loading and layout remain in their own modules.