### Back-end Overview

The Policy Browser is a web application exposing a REST\(^\text{1}\) API. The server produces data (JSON) and graphs (SVG) and sends them to the client that displays them.

The server is using Django\(^\text{2}\), a Python framework that follows the Model-View-Controller (MVC)\(^\text{3}\) architecture. The selection of Python was particularly interesting to do short prototyping iterations at the beginning of the project and the MVC architecture allowed to focus code development on the specific features of the project.

The client is using AngularJS\(^\text{4}\), an open source JavaScript framework, in conjunction with Bootstrap\(^\text{5}\), an open source toolkit for CSS development. It allowed producing a state-of-the-art GUI in a short time span.

### Back-end Graph Engine

The roles are organized into an inheritance hierarchy reflecting different levels of users expertise. For a given role, it is often necessary to visualize its “inherits from” and “inherited by” graphs.

There are various ways to draw graphs. Comparing them objectively requires defining appropriate metrics. In the case of a dependency graph (directed acyclic), a common approach is the layered graph drawing (also called Sugiyama drawing). Ideally no edge should go upward, and the goal is to minimize the number of edge crossing for readability. As for a large graph this can be a costly operation, the server relies on a dedicated graph engine, Graphviz\(^\text{6}\), for all the graphs generation.

### Front-end Detail Graph

The role is the key element granting permissions. A user may have permission granted by several roles. Sometimes it is required to know not only which resource a user is accessing, but also via which role. This is where the detail graph is useful. Since often several paths allow access to a resource, the detail graph is an effective way to represent this information.

### References

1. RBAC [https://csrc.nist.gov/Projects/Role-BasedAccess-Control](https://csrc.nist.gov/Projects/Role-BasedAccess-Control)
2. REST [https://www.w3.org/2001/sw/wiki/REST](https://www.w3.org/2001/sw/wiki/REST)
3. Django [https://www.djangoproject.com](https://www.djangoproject.com)
4. MVC [https://dl.acm.org/citation.cfm?id=50757.50759](https://dl.acm.org/citation.cfm?id=50757.50759)
5. AngularJS [https://angularjs.org](https://angularjs.org)
7. Graphviz [https://www.graphviz.org](https://www.graphviz.org)
8. SQLite [https://www.sqlite.org](https://www.sqlite.org)