QSP provider API guidelines

Author: The Hyve

Document version history

Version	Version Date	Initiator of change	Description of change
1	April 20th 2018	-	
2	May 29th 2018	Joris Borgdorff	A provider MUST provide fully functional example requests or an example client on how to call the API. (was: SHOULD)
3	June 5th 2018	Faustina Hwang, Jo Goossens	Added guidelines for non-technical description of the AP, and use cases for the API.
4	July 24th 2018	Faustina Hwang, Joris Borgdorff	- References to projects (Qualify, EIT Food) are removed - Added performance guidelines
5	October 4th 2018	Joris Borgdorff	- Added maximum response size in API guidelines

The Quisper Service Platform (QSP) is intended as a uniform platform that provides access to APIs from different providers of nutritional data, knowledge rules or personalized data. To allow uniform access to those APIs by clients, these providers should follow a set of guidelines.

The ideal solution to offering a set of webservices, is that the input and outputs of the distinct webservices are harmonized. This harmonization consists of structural (syntactic) and content (semantic) harmonization efforts. Semantic harmonization will not be pursued.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

Structural guidelines

Since the guidelines were formulated, the technologies that are generally used for these kinds of APIs have been updated. The HAL/JSON format has not seen a large uptake, and in some cases turned out to be verbose to implement. On the other hand, the Swagger and OpenAPI specifications have become more important. They allow an API provider to accurately describe what type of data an API accepts and what type of data it will returns. In addition, clients can automatically be generated from a Swagger or OpenAPI definition.

The guidelines are the following:

- Use a RESTful API (see for example https://restfulapi.net):

    - distinguish between HTTP methods (GET, POST, PUT, DELETE),
    - use HTTP status codes for meaningful feedback, and
- Uniform REST URL naming (see for example https://restfulapi.net/resource-naming/)
    - use URLs representing the content of the webservices instead of the provider names, as it will be more useful for the customers
    - Make use of a hierarchical URL rather than relying only on request contents. This clarifies the intention of a URL and simplifies the API documentation process.
    - Only use lowercase letters, numbers, and if needed hyphens.
    - Use nouns.
    - Use plurals for collections.
    - Generally avoid acronyms.
- Use JSON in requests and responses where possible. Use content negotiation headers for other data formats.
- Request and response body sizes may not exceed 10 MB (10485760 bytes).

Documentation guidelines

For users to be able to implement a client to a provider, they need enough information on how the service works.

- A provider MUST describe the purpose of the API in a non-technical summary, a non-technical description (targeted at healthcare providers, societal organisations, and professionals providing personalised nutrition advice who may want to use the service) and a technical description. The (non-)technical descriptions SHOULD describe possible uses or use cases of the API. A provider SHOULD provide an image or logo pertaining the API.
- A provider MUST describe the API with Swagger version 2.0 or OpenAPI version 3.0 (see https://swagger.io/specification/)
    - The specification MUST document all API calls, parameters and possible responses. It SHOULD contain descriptions and possible values.
    - The specification SHOULD document the structure of the requests and responses.
- A provider MUST provide fully functional example requests or an example client on how to call the API.
- A provider SHOULD provide documents or URLs of publicly available documents describing the possible contents of a request and response. For example, a reference to an ontology or thesaurus.

Security guidelines

Access to provider API's should be secure but not too hard to use for clients. QSP can connect only to providers that have the following properties:

1. The provider MUST host its API on an HTTPS connection.
2. The provider MUST use a valid SSL certificate for the HTTPS connection.
3. The provider SHOULD enable CORS by sending the header Access-Control-Allow-Origin: * with all responses.

Item 1 ensures that all data and headers are sent over an encrypted channel and cannot be read or modified. This is especially crucial for the QSP token, which should not be shared with any other party than QSP and the provider. Item 2 prevents some types of man-in-the-middle attacks, where an attacker poses as the provider to intercept all data.

A provider should accept a connection from QSP on the following conditions:

3. The provider SHOULD verify the token that QSP sends to indicate that the request originates from QSP.
4. The provider MAY inspect the HTTP 'Host' header to verify that the request originated from a QSP server.
5. The provider MAY inspect headers sent by QSP to find what user and/or client made the request.
6. The provider SHOULD NOT require additional client authentication.

In item 3, if the provider API is not public and open, the token sent by QSP allows a service to accept connections only from QSP. Item 4 consists of an additional optional verification to the same effect. The client application or QSP may send additional headers to indicate what client and user connected to the service, allowing the service to differentiate responses to different users and/or clients. Item 6 ensures that clients will not have to implement multiple security protocols to make use of a provider, making connecting to a provider more difficult. QSP should take care of the authentication so that providers do not have to. Possibly, client authentication that is part of the regular web service outside QSP can be foregone when the provider verifies that the request originates from QSP.

Performance guidelines

To give a suitable performance to clients, services should provide have a certain performance. In particular, the service should have the following characteristics.

1. Handle at least 20 requests per second.
2. Handle at least 10 simultaneous requests.
3. For every request, start a response within 29 seconds.

If the service is structured as a REST API, QSP can cache service responses. This helps achieving this performance by reducing the number of calls made to the actual service. Response caching comes at an additional cost, determined in https://aws.amazon.com/api-gateway/pricing/. It depends on the structure of the REST API how useful caching is.