Cloud Vision API Product Search

Operation

This resource represents a long-running operation that is the result of a network API call.

JSON representation

```
{
  "name": string,
  "metadata": {
    "@type": string,
    field1: ...,
    ...
  },
  "done": boolean,

  // Union field result can be only one of the following:
  "error": {
    object (Status (https://cloud.google.com/vision/product-search/docs/reference/rest/Shared.Types
  },
  "response": {
    "@type": string,
    field1: ...,
    ...
  }
  // End of list of possible types for union field result.
}
```

Fields

name	**string** The server-assigned name, which is only unique within the same service that originally returns it. If you use the default HTTP mapping, the **name** should be a resource name ending with **operations/{unique_id}**.

Fields	
`metadata`	`object` Service-specific metadata associated with the operation. It typically contains progress information and common metadata such as create time. Some services might not provide such metadata. Any method that returns a long-running operation should document the metadata type, if any. An object containing fields of an arbitrary type. An additional field "`@type`" contains a URI identifying the type. Example: `{ "id": 1234, "@type": "types.example.com/standard/id" }`.
`done`	`boolean` If the value is `false`, it means the operation is still in progress. If `true`, the operation is completed, and either `error` or `response` is available.
Union field `result`. The operation result, which can be either an `error` or a valid `response`. If `done` == `false`, neither `error` nor `response` is set. If `done` == `true`, exactly one of `error` or `response` is set. `result` can be only one of the following:	
`error`	`object (Status` (https://cloud.google.com/vision/product-search/docs/reference/rest/Shared.Types/Operation#Status)) The error result of the operation in case of failure or cancellation.
`response`	`object` The normal response of the operation in case of success. If the original method returns no data on success, such as `Delete`, the response is `google.protobuf.Empty`. If the original method is standard `Get`/`Create`/`Update`, the response should be the resource. For other methods, the response should have the type `XxxResponse`, where `Xxx` is the original method name. For example, if the original method name is `TakeSnapshot()`, the inferred response type is `TakeSnapshotResponse`. An object containing fields of an arbitrary type. An additional field "`@type`" contains a URI identifying the type. Example: `{ "id": 1234, "@type": "types.example.com/standard/id" }`.

Status

The `Status` type defines a logical error model that is suitable for different programming environments, including REST APIs and RPC APIs. It is used by gRPC (https://github.com/grpc). The error model is designed to be:

- Simple to use and understand for most users
- Flexible enough to meet unexpected needs

Overview

The `Status` message contains three pieces of data: error code, error message, and error details. The error code should be an enum value of `google.rpc.Code`, but it may accept additional error codes if needed. The error message should be a developer-facing English message that helps developers *understand* and *resolve* the error. If a localized user-facing error message is needed, put the localized message in the error details or localize it in the client. The optional error details may contain arbitrary information about the error. There is a predefined set of error detail types in the package `google.rpc` that can be used for common error conditions.

Language mapping

The `Status` message is the logical representation of the error model, but it is not necessarily the actual wire format. When the `Status` message is exposed in different client libraries and different wire protocols, it can be mapped differently. For example, it will likely be mapped to some exceptions in Java, but more likely mapped to some error codes in C.

Other uses

The error model and the `Status` message can be used in a variety of environments, either with or without APIs, to provide a consistent developer experience across different environments.

Example uses of this error model include:

- Partial errors. If a service needs to return partial errors to the client, it may embed the `Status` in the normal response to indicate the partial errors.
- Workflow errors. A typical workflow has multiple steps. Each step may have a `Status` message for error reporting.

- Batch operations. If a client uses batch request and batch response, the `Status` message should be used directly inside batch response, one for each error sub-response.
- Asynchronous operations. If an API call embeds asynchronous operation results in its response, the status of those operations should be represented directly using the `Status` message.
- Logging. If some API errors are stored in logs, the message `Status` could be used directly after any stripping needed for security/privacy reasons.

JSON representation

```
{
  "code": number,
  "message": string,
  "details": [
    {
      "@type": string,
      field1: ...,
      ...
    }
  ]
}
```

Fields

code	**number** The status code, which should be an enum value of **google.rpc.Code**.
message	**string** A developer-facing error message, which should be in English. Any user-facing error message should be localized and sent in the **google.rpc.Status.details** (https://cloud.google.com/vision/product-search/docs/reference/rest/Shared.Types/Operation#Status.FIELDS.details) field, or localized by the client.

Fields	
`details[]`	`object` A list of messages that carry the error details. There is a common set of message types for APIs to use. An object containing fields of an arbitrary type. An additional field "`@type`" contains a URI identifying the type. Example: `{ "id": 1234, "@type": "types.example.com/standard/id" }`.