API Versioning Scenarios

APIs follow an industry-standard/convention called Semantic Versioning a.k.a. Semver (reference here).

This standard states that assets should have a version number like MAJOR.MINOR.PATCH, e.g., 1.1.0, in order to understand the progression of this number, here are listed some examples:

Major Changes

An API experiences major changes when:

- An HTTP method is removed (after a deprecation notice)
- A new authentication/authorization mechanism is set in place when migrating technologies or protocols, e.g. going from basic authentication to OAuth2.0
- Removing a resource or endpoint, after a project decides to not support or expose it anymore or relocate it to a different URI/URL

In these examples, the version would change from 1.1.0 to 2.0.0.
In any case, breaking changes should be avoided wherever possible since they can cause significant disruption to your clients.

Minor Changes

The recommendation is to introduce changes in form of minor changes (unless a major change is required).
Examples of minor changes are:

- Adding a new resource/endpoint
- Supporting new HTTP methods
- Adding optional query parameters, headers, or fields in the body/data of the HTTP Request

In these examples, the version would change from 1.1.0 to 1.2.0

Patch Changes

Minor changes to the documentation of the API should be considered patch changes, going from 1.1.0 to 1.1.1

References:
https://semver.org
https://medium.com/fiverr-engineering/major-minor-patch-a5298e2e1798
https://thoughtspile.github.io/2021/11/08/semver-challenges/