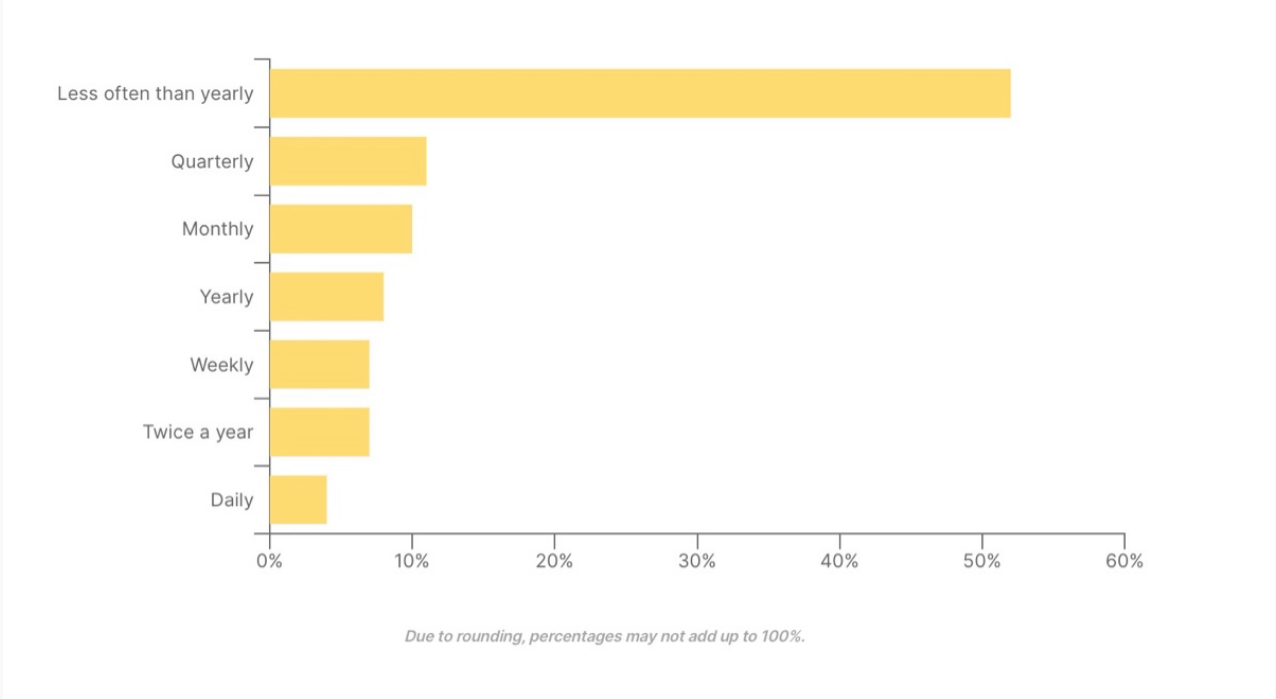
Risk Assessment — Tonic API

⚠️ State of API security

According to the Postman 2022 State of the API report, security remains a very present concern for API developers. While 52% of respondents to the research conducted by Postman said that API security incidents happened less than once a year (see graph below), 20% reported incidents that happened at least once monthly. This number is sizeable, which highlights just how important it is to focus on security when we work with APIs.



Graph taken from the Postman 2022 State of the API report.

In the following tables, then, I have outlined some common security risks that might occur when working with an API such as Tonic. I believe it is important to have a perspective of the potential risks from the very beginning of the development process, and to incorporate this awareness into the design process from the very beginning of the API lifecycle.

That said, as the Tonic API is a project that is still in its very early stages, I do not feel that I have the software architecture knowledge to offer a list of best practices and guidelines to prevent these potential risks. I do, however, understand the importance of incorporating testing into the software design process from the very beginning, to hopefully eliminate security breaches before they even happen. A good place to get a clear idea of these design principles is Robert C. Martin's chapter 28 (The Test Boundary) in his book *Clean Architecture*. Another resource I will often be referencing in regards of risks and security is the Open Web Application Security Project's documentation.

⚠️ Rating criteria

The development of the ratings is based on a) the likelihood of an event occurring (from most unlikely to most likely) and b) the severity of the issues that might arise if the event does occur (from trivial issues to major issues). This risk assessment is heavily based on the risk assessment template available on Confluence, which I have used frequently to create documentation and record progress on this project.

LOW	MEDIUM	HIGH	EXTREME
· Acceptable · Ok to proceed, issue warning	· Take mitigation efforts	· Generally unacceptable · Seek support	· Intolerable · Place event on hold

---

⚠️ Assessment for the Tonic API (ver. 1)

SEVERITY ➡️ LIKELIHOOD ⬇️	ACCEPTABLE Little/no effect	TOLERABLE Effects present but not critical	UNDESIRABLE Serious impact	INTOLERABLE Could result in disaster
IMPROBABLE _Risk is unlikely to occur_	No acceptance testing	Lack of zero-trust policy		Incorrectly implemented authentication
POSSIBLE _Risk will likely occur_	Insufficient unit testing	Not enough constraints on data	Injections, insufficient integration tests	Lack of data encryption (missing TLS)
PROBABLE _Risk will occur_	Duplicate records in database	Weak software architecture	Denial of Service (use HTTPS, limit requests)	Lack of logs/ monitoring (detecting breaches when it's too late)