1. Free Software implementation

GNU Taler must be Free Software. For merchants, our Free Software reference implementation prevents vendor lock-in. As the software of the payment provider itself is free, countries can deploy the payment system without compromising sovereignty.

Customers benefit from Free Software as the wallet software can be made to run on a variety of platforms, and the absence of user-hostile features such as tracking or telemetry can easily be assured.

2. Protect the privacy of buyers

Privacy should be guaranteed via technical measures, as opposed to mere policies. Especially with micropayments for online publications, a disproportionate amount of rather private data about buyers would be revealed, if the payment system does not have privacy protections.
In legislations with data protection regulations (such as the recently introduced GDPR in Europe), merchants benefit from this as well, as no data breach of customers can happen if this information is, by design, not collected in the first place. Obviously some private data, such as the shipping address for a physical delivery, must still be collected according to business needs.

3. Enable the state to tax income and crack down on illegal business activities

As a payment system must still be legal to operate and use, it must comply with these requirements. Furthermore, we consider levying of taxes as beneficial to society.

4. Prevent payment fraud

This imposes requirements on the security of the system, as well as on the general design, as payment fraud can also happen through misleading user interface design or the lack of cryptographic evidence for certain processes.

5. Only disclose the minimal amount of information necessary

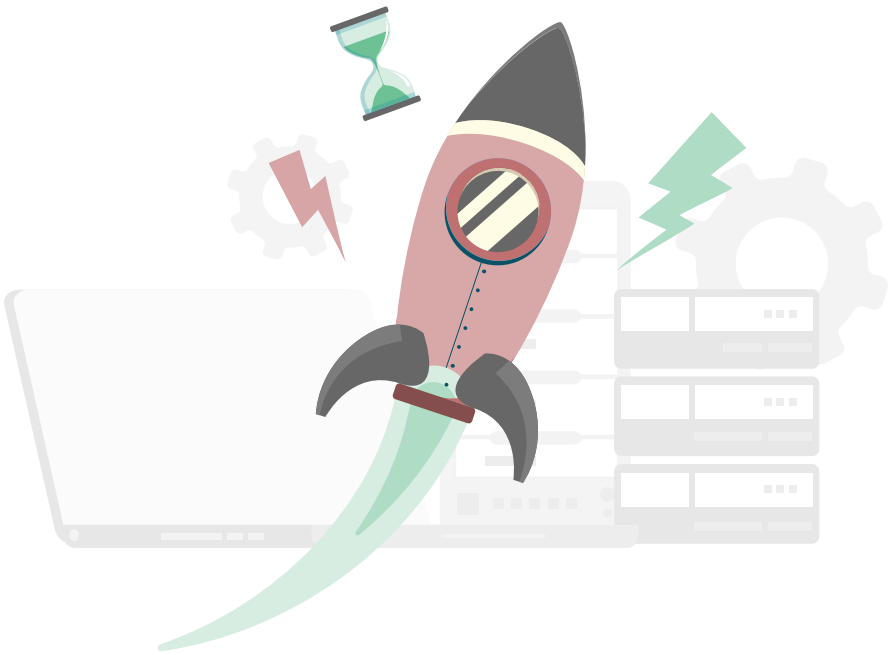
The reason behind this goal is similar to (2). The privacy of buyers is given priority, but other parties such as merchants still benefit from it, for example, by keeping details about the merchant's financials hidden from competitors.

6. Be usable

Specifically it must be usable for non-expert customers. Usability also applies to the integration with merchants, and informs choices about the architecture, such as encapsulating procedures that require cryptographic operations into an isolated component with a simple API.

7. Be efficient

Approaches such as proof-of-work are ruled out by this requirement. Efficiency is necessary for GNU Taler to be used for micropayments.



8. Fault-tolerant design

Taler should tolerate failure of individual components and systems, including malicious operators compromising core secrets. This manifests in architectural choices such as the isolation of certain components, and auditing procedures.

9. Foster competition

It must be relatively easy for competitors to join the systems. While the barriers for this in traditional financial systems are rather high, the technical burden for new competitors to join must be minimized. Another design choice that supports this is to split the whole system into smaller components that can be operated, developed and improved upon independently, instead of having one completely monolithic system.