# Why Are Abstractions Important?

We'll cover the following ⌃

- What is abstraction?
- Database abstraction
- Abstractions in distributed systems

What is abstraction?

Abstraction is the art of obfuscating details that we don't need. It allows us to concentrate on the big picture. Looking at the big picture is vital because it hides the inner complexities, thus giving us a broader understanding of our set goals and staying focused on them. The following illustration is an example of abstraction.
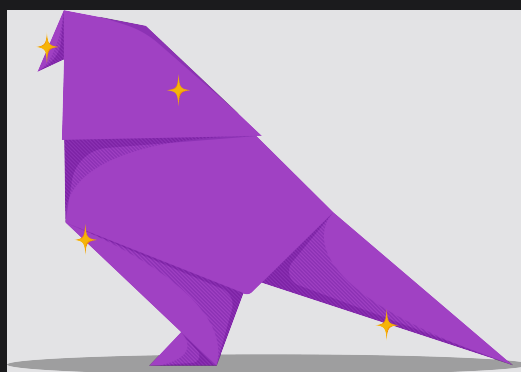
Abstraction of a bird

With the abstraction shown above, we can talk about birds in general without being bogged down by the details.

> **Note:** If we had drawn a picture of a specific bird or its features, we wouldn't achieve the goal of recognizing all birds. We'd learn to recognize a particular type of bird only.

In the context of computer science, we all use computers for our work, but we don't start making hardware from scratch and developing an operating system. We use that for the purpose at hand rather than digging into building the system.

The developers use a lot of libraries to develop the big systems. If they start building the libraries, they won't finish their work. Libraries give us an easy interface to use functions and hide the inside detail of how they are implemented. A good abstraction allows us to reuse it in multiple projects with similar needs.

Database abstraction

Transactions is a database abstraction that hides many problematic outcomes when concurrent users are reading, writing, or mutating the data and gives a simple interface of commit, in case of success, or abort, in case of failure. Either way, the data moves from one consistent state to a new consistent state. The transaction enables end users to not be bogged down by the subtle corner-cases of concurrent data mutation, but rather concentrate on their business logic.

Abstractions in distributed systems

Abstractions in distributed systems help engineers simplify their work and relieve them of the burden of dealing with the underlying complexity of the distributed systems.

The abstraction of distributed systems has grown in popularity as many big companies like Amazon AWS, Google Cloud, and Microsoft Azure provide distributed services. Every service offers different levels of abstraction. The details behind implementing these distributed services are hidden from the

users, thereby allowing the developers to focus on the application rather than going into the depth of the distributed systems that are often very complex.

Today's applications can't remain responsive/functional if they're based on a single node because of an exponentially growing number of users. Abstractions in distributed systems help engineers shift to distributed systems quickly to scale their applications.

> **Note:** We'll see the use of abstractions in communications, data consistency, and failures in this chapter. The purpose is to convey the core ideas, but not necessarily all the subtleties of the concepts.