Prior to an organization delivers a product, the engineering team needs to decide on the most suitable application architecture. In most of the cases 2 distinct models are referenced: monoliths and microservices. Regardless of the adopted structure, the main goal is to design an application that delivers value to customers and can be easily adjusted to accommodate new functionalities.

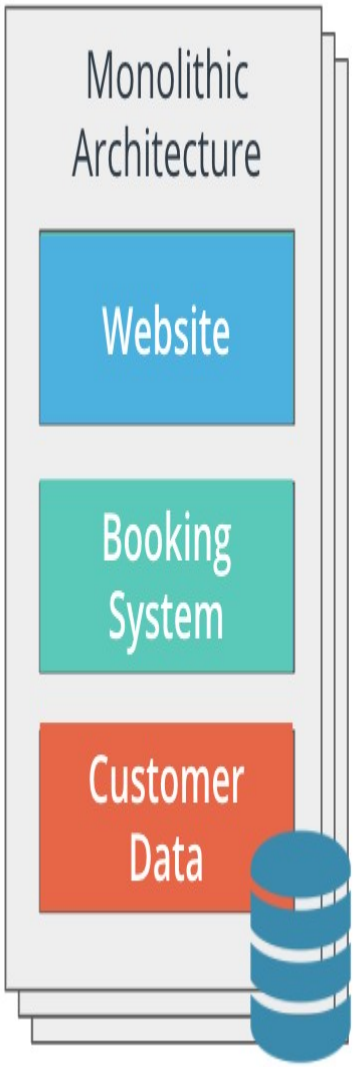
Also, each architecture encapsulates the 3 main tires of an application:

- UI (User Interface) - handles HTTP requests from the users and returns a response
- Business logic - contained the code that provides a service to the users
- Data layer - implements access and storage of data objects

Monoliths

In a monolithic architecture, application tiers can be described as:

- part of the same unit
- managed in a single repository
- sharing existing resources (e.g. CPU and memory)
- developed in one programming language
- released using a single binary



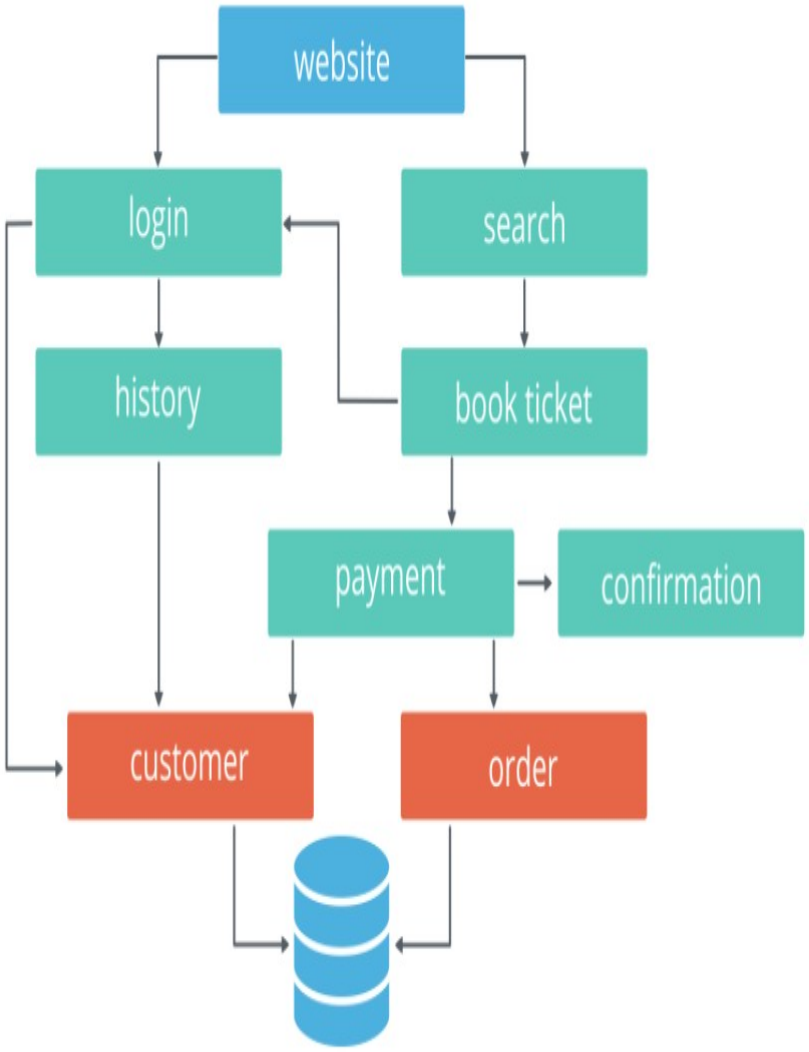
A booking application referencing the monolithic architecture

Imagine a team develops a booking application using a monolithic approach. In this case, the UI is the website that the user interacts with. The business logic contains the code that provides the booking functionalities, such as search, booking, payment, and so on. These are written using one programming language (e.g. Java or Go) and stored in a single repository. The data layer contains functions that store and retrieve customer data. All of these components are managed as a unit, and the release is done using a single binary.

Microservices

In a microservice architecture, application tiers are managed independently, as different units. Each unit has the following characteristics:

- managed in a separate repository
- own allocated resources (e.g. CPU and memory)
- well-defined API (Application Programming Interface) for connection to other units
- implemented using the programming language of choice
- released using its own binary



A booking application referencing the microservice architecture

Now, let's imagine the team develops a booking application using a microservice approach.

In this case, the UI remains the website that the user interacts with. However, the business logic is split into smaller, independent units, such as login, payment, confirmation, and many more. These units are stored in separate repositories and are written using the programming language of choice (e.g. Go for the payment service and Python for login service). To interact with other services, each unit exposes an API. And lastly, the data layer contains functions that store and retrieve customer and order data. As expected, each unit is released using its own binary.

New terms

- **Monolith**: application design where all application tiers are managed as a single unit
- **Microservice**: application design where application tiers are managed as independent, smaller units

Further reading

- What’s the Difference Between Monolith and Microservices?
- Microservices vs Monolithic Architecture