<https://ryax.tech/how-to-build-backend-part1/>

<https://www.cosmicpython.com/>

TODO:  
1. Choose a design.

2. Choose a framework.

3. Create example project.

4. Create diagrams.

architecture composed of 3 layers: Domain, Application, and Infrastructure.

1. The Domain layer defines the Data structures in plain Python objects: the business objects.
2. The Application layer holds the brain of the App: the business logic.
3. Finally, the Infrastructure layer is the "arms and legs" of our App: the part that interacts with the external world (HTTP API, database, file system, servomotors, etc).

Continue at -

# Dependencies injection

The goal of dependency injection is to avoid creating objects everywhere or passing them in all functions in some kind of Context melting pot. To do so we'll define where all Infrastructure services are created, in one single place. We can then easily inject these services as dependencies of Application services using a default value as a singleton (e.g. for a database connection) or a one-time object from a factory (e.g. for an HTTP request handler).

The [**Dependency Injector**](https://python-dependency-injector.ets-labs.org/) library is well designed and provides everything you need to define all your services, inject them and even load configurations.

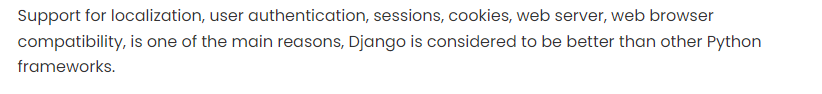
# Frameworks to consider

<https://www.monocubed.com/blog/top-python-frameworks/>

<https://www.hug.rest/>

Flask

Django



FastAPI