Web Services

* a service is, essentially, software that provides services to other components over a network
* a web service is the above, where the network is the internet
  + follows service-oriented architecture (SOA)
    1. logically represents a business activity with a specified outcome
    2. self-contained
    3. black box for consumers (they don't know the details of their interaction with the software)
    4. may consist of underlying services (doesn't have to)
  + two main types
    1. **SOAP**: simple object access protocol. older, more rigid design, transfers XML documents, any protocol
    2. **REST**: representational state transfer. newer, more flexible, transfers any type of document, protocol must be HTTP

REST

* Representational State Transfer
* a web service architecture designed by Roy Fielding
* transfer the state of a representation of an object to a database
* a RESTful web service must meet the following constraints:
  1. client-server architecture: the front end and back end of the application are two separate applications (i.e. Java back end, Angular/React/etc. front end)
  2. statelessness: the server does not store client context between requests, all necessary information is sent to the server in every request from the front end
  3. cacheability: particular responses from the back end are able to be cached for easy reuse, especially if they are used often
  4. layered system: the user cannot tell whether there are any intermediaries between them and the database
  5. code on demand: OPTIONAL. where necessary, the back end can send scripts to offer additional functionality to the front end as a response.
  6. uniform interface: requests/responses use the following patterns:  
     **i.** **URI model**: requests are sent in a particular format: /collection/identifier  
     **ii.** resources are manipulated through representations, i.e. when changes are made, they are made to the representations being sent, rather than in the back end after certain directions were given.  
     **iii.** self-descriptive messages: a GET request to "/cat" should get the cats. **iv.** HATEOAS: hypermedia as the engine of application state.
     + when a resource has a lot of information, it can include URIs so that the client can send requests to those URIs for further information.