Micro Service

* Easier maintenance:
  + Each service is independent of each other, so when something goes wrong in a service, we can go the specific service and figure out the problem
* Separation of concern:
  + The key development concept that we want one code to do one thing well. Leading to each service will focus on one thing and do it well
* Different language / platform / OS:
  + Each program language has their own advantages. For example, one language might be very good at performing a specific calculation, and others may not.
    - Java: PRO, it does automatic “*garbage collecting*”, automatically free up the space of a resource when we exit the scope of the resource. CON, because of automatic garbage collecting, it runs slower comparing to other languages like C++
      * main() {int answer = multiply(3, 7);}
      * public int multiply(int a, int b){int result = a \* b; return result;}
      * in above case, we enter from main, first thing it does it allocate a memory space for answer, and when it enters to multiply function, we go in to the function signature and body, it allocate another memory space for result, calculate a \* b (21) and store it in result, then it goes back to main and store 21 (again) to answer. The garage here is the memory resource for variable result. Because we exited out the function, therefore, we don’t need the variable anymore so the memory space needs to free up, and Java does it automatically for you.
    - Java: it is widely used programming language, therefore there is a big community.
    - C++: PRO, it is faster because it doesn’t do automatic garbage colleting. CON, it makes developers more aware when it comes to decision of garbage collecting.
* Robustness
* Scalable
* Easy to deploy
* Reusable and replaceable

API (do more research on this along with REST)

* Application Programming Interface
  + In short, an interface to interact with your program
  + Example: