MICROSERVIZI  
  
**I microservizi sono un’architettura software** in cui un applicazione viene suddivisa in piccoli servizi modulare e indipendenti e ognunoe responsabile di una specifica funzionalità.  
  
Quindi di seguito vediamo come funzionano queste applicazioni e come vengono esposti e consumati i servizi:   
 1) Sviluppo dei microservizi → I microservizi vengono sviluppati come componenti separati, ognuno ha la propria logica di buisness e persistenza dei dati. Spesso vengono utilizzate librerie e framework come spring boot per semplificare la gestione dei microservizi.  
 2) **Eposizione dei servizi** → Ogni microservizio espone il suo servizio tramite un’interfaccia ben definita, **spesso come un API’RESTful**.  
L’API definisce i punti di ingresso (end-point) i quali sono gestiti dai controller e attraverso i quali gli altri servizi o applicazioni possono accedere alle funzionalità del microservizio.  
 3) **Comunicazione tra i microservizi** → la comunicazione tra i vari microservizi è fondamentale. Uno dei modi più comuni per gestire la comunicazione tra i microservizi è tramite HTTP/HTTPS utilizzando richieste RESTful.  
 4) **Consumo dei servizi** → Nella parte front-end i servizi vengono consumati invocando le API esposte dai microservizi. **Angular può effettuare richieste HTTP ai relativi endpoint** e gestire le risposte ottenute e le chiamate possono essere sincrone o asincrone.

In un’applicazione spring, possiamo effettuare chiamate **API REST tramite HTTP,** l’acronimo REST sta per   
**REpresentational state transfer,** Rest viene utilizzato indipendentemente dal tipo di linguaggio.  
Prima di effettuare una richiesta HTTP devo esporre degli end-point.  
  
Chiaramo il concetto di richieste RESTful e APR RESTful  
 1) **le richieste RESTful** sono un tipo di richieste che seguono i principi dell’architettura REST.  
Ricordiamo che REST è un approccio architetturale per la progettazione di servizi web e **RESTful si riferisce ad un’implementazione di servizi web basata su queste linee guida.**Le richieste RESTful sono inviate dai client (quindi chiamate effettuate dai client, come ad’esempio un applicazione front-end) verso un servizio RESTful o ai server per accedere ad una base di dati per creare,aggiornare,eliminare o leggere (risorse,dati,record) e quindi per eseguire azioni offerte dall’API RESTful   
  
**Le richieste RESTful vengono eseguite utilizzando il protocollo HTTP, che offre diversi metodi di richiesta come GET,POST,PUT,DELETE corrispondenti alle operazioni CRUD le quali consentono la manipolazione delle risorse.**  
2) Mentre le API RESTful forniscono un'interfaccia standardizzata attraverso cui i client possono interagire con il server, queste sono insiemi di end-point o punti d’ingresso definiti da un servizio web RESTful i quali endpoint rappresentano le operazioni che il servizio web può svolgere sulle risorse nel database, **consentendo quindi di eseguire operazioni CRUD e quindi di interagire con il servizio web o sito.  
Le operazioni CRUD vengono eseguite su determinate risorse, come dati o oggetti, e queste risorse sono identificate appunto dagli end-point che tramite URL rendono accessibi  
  
Quindi l’obiettivo principale delle API RESTful è consentire a client diversi di interagire con il servizio web tramite l’esposizione di end-point.**in sintesi possiamo dire che: Le chiamate RESTful vengono fatte dai client ai server per poter accedere a risorse le quali sono accessibili tramite le API RESTful che sono mappate su un determinato end-point e quindi un url specifico per poter eseguire le operazioni CRUD

**Generalemente i parametri delle richieste vengono trasmessi tramite URL**, le risposte possono essere inviate in diversi formati come JSON o XML  
sia la APIRESTful che le richieste RESTful sono componenti chiave di unarchitettura REST  
  
Seconda sintesei che può aiutare a capire:  
 **Tutto il meccanismo funziona che si costruisce un API, si costruisce un servizio, si fornisce un endpoint per l’accesso al servizio e si forniscono i dati in base alle richieste.**