**Was ist Rest(ful) API:**

Programmschnittstelle (API), die HTTP-Anfragen für den Zugriff auf und die Verwendung von Daten verwendet.

Datentypen GET, PUT, POST und DELETE verwenden, Abrufen, Aktualisieren, Erstellen und Löschen von Operationen in Bezug auf Ressourcen.

Eine API für eine Website ist Code, der es zwei Softwareprogrammen ermöglicht, miteinander zu kommunizieren.

RESTful APIs können mit JavaScript oder Python erstellt werden.

**Wie funktioniert es:**

RESTful API verwendet bestehende HTTP-Methoden, die durch das RFC-2616-Protokoll definiert sind. Dazu gehören:

1. GET, um eine Ressource abzurufen;
2. PUT, um den Zustand einer Ressource (die ein Objekt, eine Datei oder ein Block sein kann), zu ändern oder zu aktualisieren;
3. POST, um diese Ressource zu erstellen, und
4. DELETE, um sie zu entfernen.

**Dateiformate sind:**

1. .json müssen
2. .xml
3. .x-wbe+xml
4. .x-www-form-urlencoded
5. .form-data

**Beschränkungen:**

Verwendung einer einheitlichen Schnittstelle: Ressourcen durch eine einzige URL eindeutig identifizierbar sein.

Client-/Server: Es sollte eine klare Abgrenzung zwischen Client und Server geben.

Zustandslose Operationen: Alle Client-Server-Operationen sollten zustandslos/stateless sein (d.h. es keine Aufzeichnungen über frühere Interaktionen gibt), und jede erforderliche Zustandsverwaltung sollte auf dem Client und nicht auf dem Server stattfinden.

RESTful-Ressourcen-Caching: Alle Ressourcen sollten eine Zwischenspeicherung ermöglichen.

Mehrschichtiges System. REST ermöglicht eine Architektur, die aus mehreren Schichten von Servern besteht.

Code auf Anfrage. Die meiste Zeit sendet ein Server statische Darstellungen von Ressourcen in Form von XML oder JSON zurück. Bei Bedarf können die Server jedoch auch ausführbaren Code an den Client senden.

**Herausforderungen:**

* Das design und die Architektur.
* Konsistenz der Endpunkte: die Pfade müssen konsistent sein. (Gemeinsame Webstandards, d.h. es hat Regeln und Strukturen von Html, Css, JavaScript und Svg(Grafiken))
* API-Versionierung: Endpunkt-URLs müssen immer gültig sein.
* Lange Antwortzeit: Mit der Zeit, könnte es länger brauchen
* Viele Daten: Mit der Zeit, häufen sich daten an.
* Security: Validieren von URLs und blockieren von unbekannten IP-Adressen, Domänen und unerwartete grosse Nutzerdaten.