# Rest Schnittstelle

Eine REST (= Representational State Transfer) Schnittstelle, auch REST API (=Applications Programming Interface) genannt, ermöglicht Web-Anwendungen, HTTP-Zugriffsmethoden zu benutzen. Viele Web-Anwendungen sind in Form von Webservices, weil man damit alle Arten von Clients bedienen kann. In einem Projekt, dass zum Beispiel aus einer WPF Applikation, einer einfachen Website und Mobile Apps für, iOS, Android und Windows besteht, kann man über dieselbe Rest Schnittstelle zugreifen. Rest Schnittstellen die für andere Softwareentwickler zur Verfügung gestellt werden, müssen dokumentiert werden. Damit Entwickler von Clients wissen welche Zugriffe möglich sind.

REST APIs basieren nur auf eine Datenbank. Bei Rest APIs kann das Datenaustauschformat JSON oder XML sein.[[1]](#footnote-1) Das Projekt Jast benutzt JSON als Datenaustauschformat.

## Rest Services

Webservices werden nach dem REST Schema erstellt. Bei REST Services, häufig auch REST API, können verschiedene HTTP- Zugriffsmethoden angewandt werden, dabei werden die CRUD-Funktionen (Erstellen, Lesen, Aktualisieren, Löschen) für Ressourcen bereitgestellt. Auf der Serverseite werden meistens die Ressourcen als Datensätze in einer rationalen Datenbank gespeichert.[[2]](#footnote-2)

**Die CRUD-Funktionen und ihre Aufgaben**

|  |  |  |
| --- | --- | --- |
| HTTP Methode | Aufgabe | Erklärung |
| GET | Lesen | Liefert die gewünschten Informationen der Ressourcen am Server. |
| POST | Erstellen von neue Ressourcen | Die Daten im Body werden auf dem Server neuangelegt. |
| PUT | Ändern von vorhandene Ressourcen. | Die Ressourcen vom Server werden an die Daten im Body angepasst. |
| DELETE | Löschen | Löscht Daten vom Server. |
| OPTIONS |  |  |
| HEAD |  |  |
| TRACE |  |  |
| CONNECT |  |  |

Tabelle 1 HTTP Methoden[[3]](#footnote-3)

**Zulässige Zugriffsmethoden und Pfade von Jast:**

|  |  |  |
| --- | --- | --- |
| HTTP Methode | Pfad | Funktion |
| GET | /JAST/rest/users/ | Liefert die Liste der User. |
| GET | /JAST/rest/users/ID | Liefert die User mit der angegebenen numerischen ID. |
| POST | /JAST/rest/users/ | Erstellt einen neuen User mit den im Body angegebenen Daten. |
| PUT | /JAST/rest/users/ID | Ersetzt die Daten vom User mit der angegebenen ID durch diejenigen im Body. |
| DELETE | /JAST/rest/users/ID | Löscht die User mit der angegebenen ID. |
| GET | /JAST/rest/quizes/ | Liefert die Liste der Quizze. |
| GET | /JAST/rest/quizes/ID | Liefert die Quizze mit der angegebenen numerischen ID. |
| GET | /JAST/rest/quizes/category/ID | Liefert die Quizze mit der Kategorie der angegebenen numerischen ID. |
| GET | /JAST/rest/content/ | Liefert die Liste der Quizinhalte. |
| GET | /JAST/rest/content/ID | Liefert die Liste der Quizinhalte mit der angegebenen numerischen ID. |
| GET | /JAST/rest/content/quiz/ID | Liefert die Quizinhalte vom Quiz mit der angegebenen numerischen ID. |

Tabelle 2 Überblick über die Methode der REST-API von Jast

//Noch nicht alle Pfade

## Rest Client

REST-APIs können nicht in einem Browser durch ein Link wie Webanwendungen getestet werden. Mit Hilfe von REST-Clients kann man die REST-Schnittstelle testen. Fertige Rest Clients können von anderen Software Entwicklern, die die REST-Schnittstelle brauchen, als Referenzimplementierung benutzt werden. Hier können dann die REST-Methoden getestet werden. [[4]](#footnote-4)

### Postman

Postman ist ein REST-Client das von Google Chrome kostenlos zur Verfügung gestellt wird. Die Erweiterung Postman kann einfach im Browser installiert werden.

**Aufbau von Postman:**

1. Hier können die HTTP-Methoden ausgewählt werden.
2. Hier wird der vollständige URL eingetragen. Man muss z.B. <http://vm86.htl-leonding.ac.at:8080/JAST/rest/users> als URL eintragen, wenn man die User bekommen möchte. Mit einem? hinter der URL können sie weiter String-Abfragen eingeben. Params ermöglicht die Eingabe der weiteren String-Abfragen in Form einer Tabelle.
3. Hier kann man die Authorization auswählen, wenn eine vorhanden ist.
4. Bei einem PUT und POST Request können hier die Ressourcen eingegeben werden. Dabei kann ausgewählt werden in welchem Format die Eingabe erfolgt. Bei raw kann man auch das Textformat auswählen. In unserem Fall muss JSON ausgewählt werden. Hiermit wird der Header gleich angepasst.
5. Abfrage abschicken.
6. Status zeigt an, wie die Abfrage abgelaufen ist.
7. Hier wir die Antwort vom REST-Server angezeigt.
8. Verlauf der ausgeführten Abfragen.

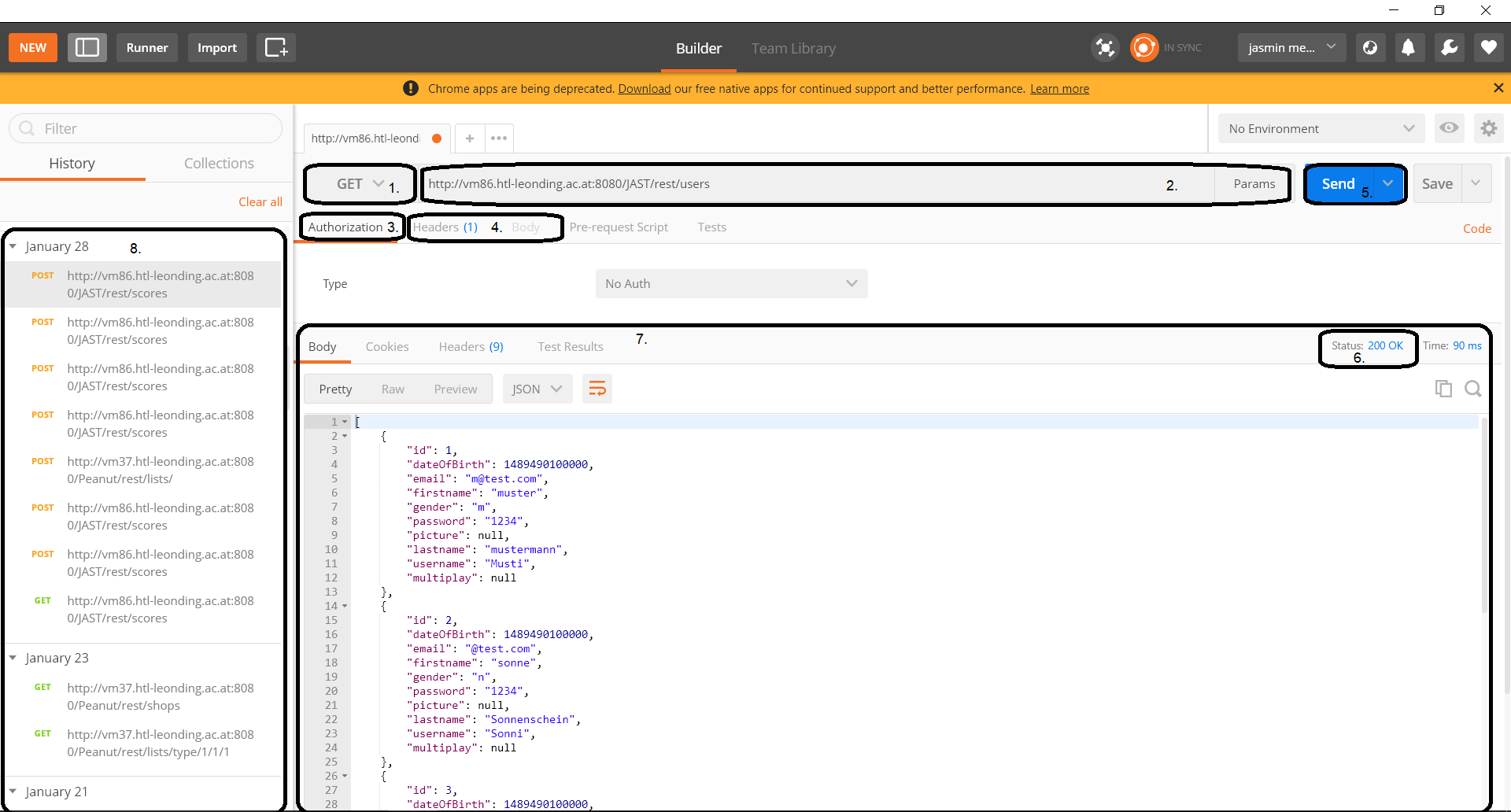


Abbildung 1 Rest Client Postman

## Sicherheit

<http://www.searchenterprisesoftware.de/tipp/Wie-sich-REST-API-Endpunkte-fuer-Cloud-Anwendungen-absichern-lassen>

### CORS (Cross-Origin Resource Sharing)

//folgt [file:///C:/Users/User/Downloads/Bachelor-Thesis\_Julian\_Suleder\_181348.pdf]

### OAUTH

OAUTH ist ein weit verbreiteter Sicherheitsstandard, dass einer Softwareanwendung den Zugriff auf geschützte Ressourcen auf einer Web-Schnittstelle ohne sich auszugeben, ermöglicht. Dafür fordert die Anwendung zuerst die Autorisierung des Besitzers. Dies erfolgt mithilfe von Token. Der Zugriff auf ein REST Service ist bei einem OAUTH Autorisierungsframework begrenzten. [[5]](#footnote-5)

//Genauer?

<https://www.thalia.de/shop/home/suchartikel/it_handbuch_fuer_fachinformatiker/sascha_kersken/EAN9783836244268/ID55430857.html>

1. (**vgl**. Name 2017, S.1099 f.)  [↑](#footnote-ref-1)
2. (**vgl**. Name 2017, S.1099 f.)  [↑](#footnote-ref-2)
3. (**vgl**. Name 2017, S.1099)  [↑](#footnote-ref-3)
4. (**vgl**. Name 2017, S.1120)  [↑](#footnote-ref-4)
5. (**vgl**. Name Jahr, S.3 ff.)  [↑](#footnote-ref-5)