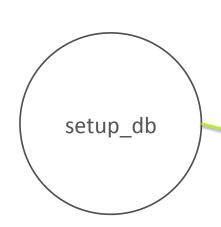
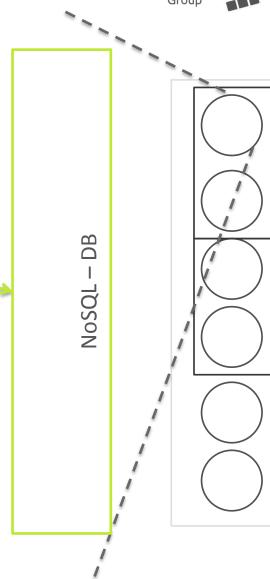
# > Architecture



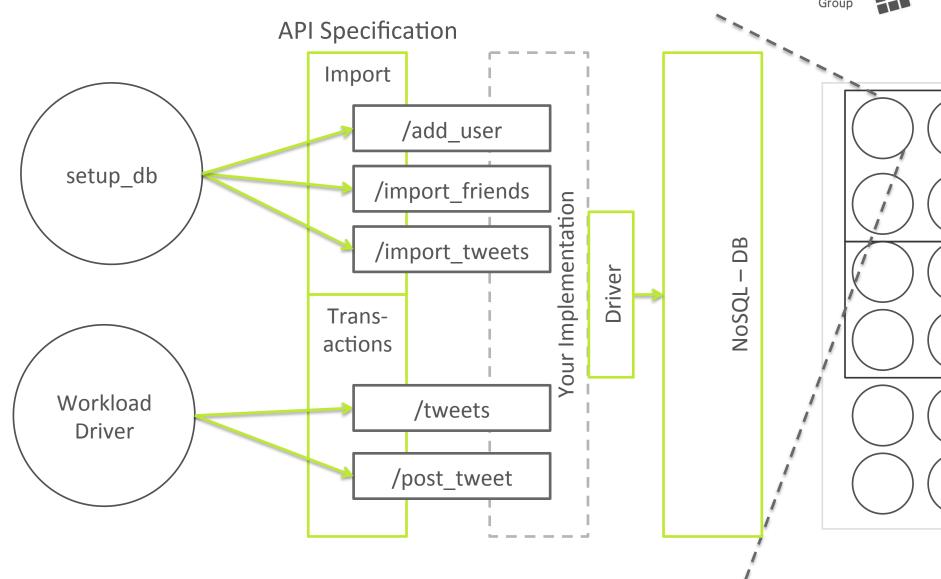






## > Architecture





## **HTTP-Post Requests**

JSON-encoded payload

#### Queries

- /tweets {"username": "wl", "start":0, "limit": 40} [Primär]
- /post\_tweet {,,username": ,,wl", ,,body": ,,Schon wieder WSDM lesen ..."}
- /create\_user {,,username": ,,wl", ,,password": ,,xyz"}
- /add friend {",username": ",wl", ",friend": ",dh"}
- / (Public Timeline)

## Für Datenbank-Setup

- /import\_friends {,,username": ,,wl", ,,friends": [,,dh", ,,mt"]}
- /import\_tweets {"username": "wl", "bodies": ["<Text>", "<Text>"], "timestamps": [23143252, 2323432]}
- /delete\_tweets\_afer{,,delete\_afer ": 2323432}

## Nächste Schritte



#### Bis Nächste Woche

- Schemakonzeption
  - Welche Daten?
  - Wie werden sie in eurer DB abgelegt?
  - Welche Anfragen müssen gestellt werden (zwischen DB und Applikation)
  - Das Schema sollte schon in Richtung der Timeline-Anfrage getunt sein
- Technische Voraussetzung: REST Apis
  - z.b. Flask in Python
  - Play in Java (oder Spark)
  - Sinatra in Ruby
  - Express in Node.js
  - etc
  - pp
- Technische Voraussetzung: JSON Library
  - z.b. GSON in Java
- Präsentation:
  - 1-3 Folien zur Schema und Anfragekonzeption