FRIEDRICH-SCHILLER-
UNIVERSITÄT
JENA


(GraphQL 2023)

Natascha Stumpf

# Graphdatenbank - GraphQL

14.09.2023

# Gliederung

- **GraphQL**
  - Überblick
  - User & Community
- **Tutorial**
  - Einleitung
  - Environment
  - Python
  - Datenbank
  - Apollo Explorer
  - Fazit
  - WAS FEHLT?
- **Referenzen**

# GraphQL - Überblick

- ➢ Open-Source-Datenabfrage- und –Bearbeitungssprache für APIs und Abfrage-Laufzeit-Engine

- Nur ein entry point

- Front-End-Entwickler beschreiben die gewünschten Daten
- ➢ Daten können einfach gefunden und angefordert werden

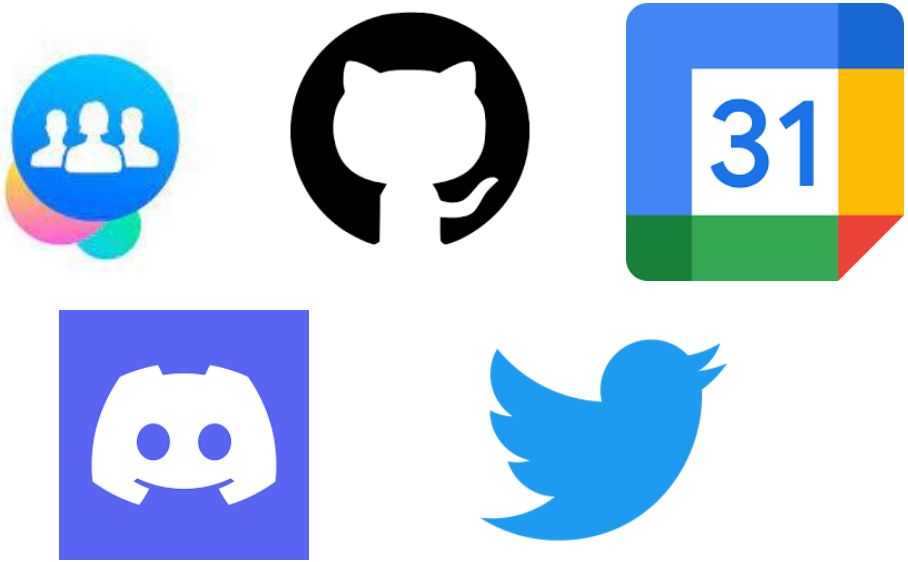
- Backend-Entwickler schreiben Code, um die Anfragen zu lösen
- ➢ Stellen das Schema für die Daten auf (schema.gql)
- ➢ Syntax in jeder Programmiersprache

(GraphQL 2023)

# GraphQL – User & Community

**User**


**Community**




(GraphQL 2023)

# Tutorial – Einleitung

## Zielsetzung

- Einrichten eines Python-Webservers mit Flask
- Ariadne-Bibliothek zur Implementierung von GraphQL
- GraphQL-Schema zusammenstellen
- Abfragen und Mutationen gegen eine Python GraphQL API ausführen

(Haque 2023)

# Tutorial – Environment

- Flask

Erstellung der Webanwendung der API

- Flask-SQLAlchemy

Um auf SQL-Datenbank zuzugreifen und sie in Python-Anwendung einzubinden

- Ariadne

Integration von GraphQL in Python-Anwendungen

- Flask-Cors

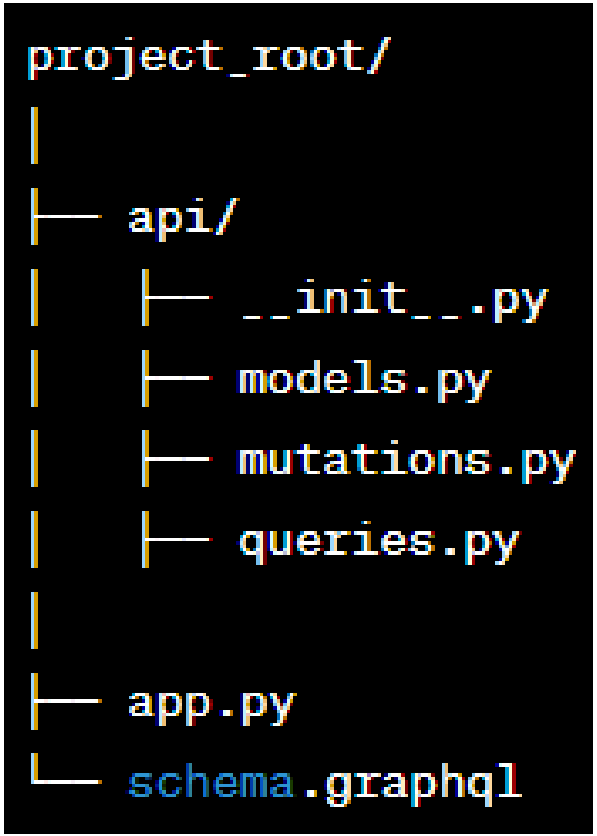
Kommunikation Webanwendungen auf unterschiedlichen Domänen, Zugriff auf Ressourcen anderer Domänen gewähren

(Flask 2023; Ariadne 2023)

# Tutorial – Python

**api/__init__.py**

Enthält gesamten API-bezogenen Konfigurationscode




(Haque 2023)

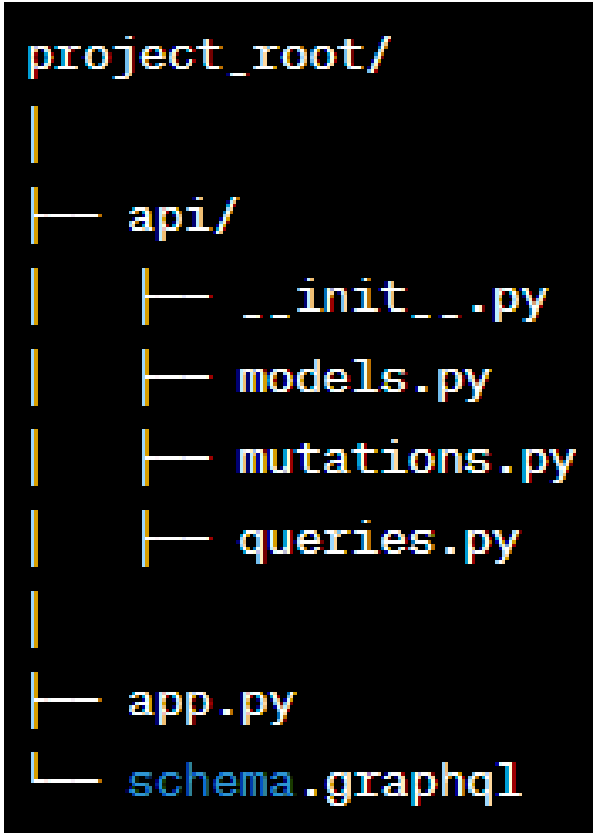
# Tutorial – Python

**app.py**

für den eigentlichen Start der Flask-Anwendung verantwortlich

**To start (in conda)**
conda activate
cd Arbeitsordner/
Set FLASK_APP=app.py
Eco %FLASK_APP%
Flask run

**Browser**
http://127.0.0.1:5000/




(Haque 2023)

# Tutorial - Datenbank

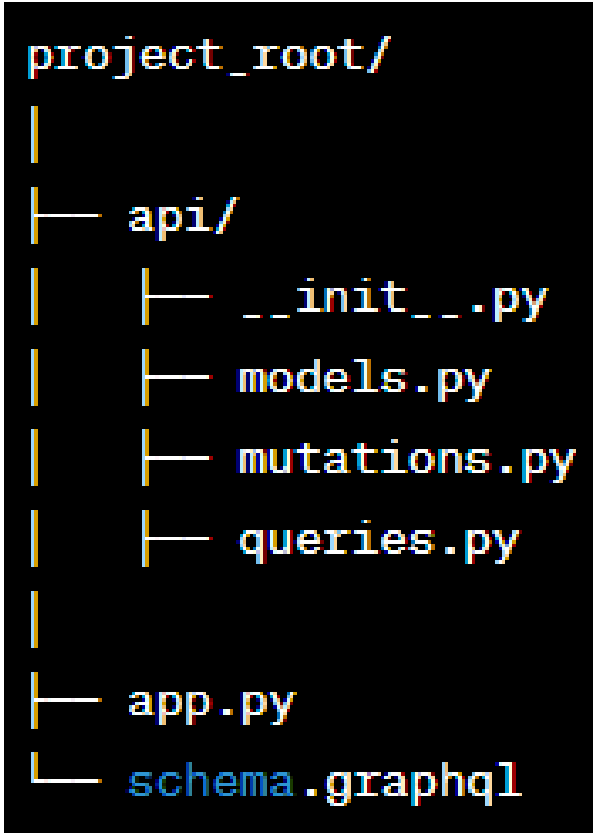
**PostgreSQL**

- Datenbank hinzufügen

**api/__init__.py**

Anpassen: YourUserName, YourPassword, YourHostname, YourDatabaseName

```
app.config["SQLALCHEMY_DATABASE_URI"] =
"postgresql://YourUserName:YourPassword@Y
ourHostname:5432/YourDatabaseName"
```

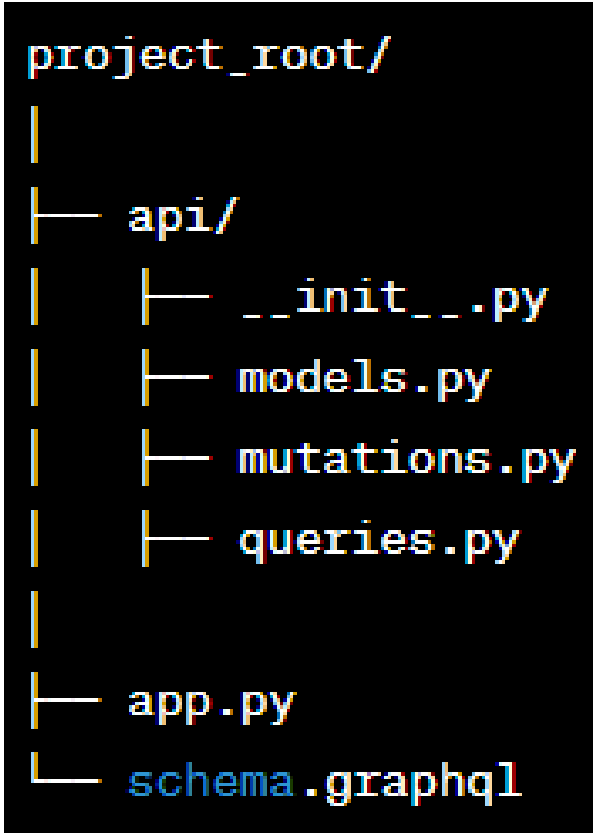



(Haque 2023)

# Tutorial - Python

**api/models.py**

- Post-Tabelle erstellen
- Klasse Post
- Ein Beitrag hat
  - eine eindeutige ID
  - einen Titel
  - eine Beschreibung
  - das Erstelldatum




(Haque 2023)

# Tutorial – Python

**schema.graphql**

- GraphQL einbinden
- Post-Tabelle erstellen
- Ein Beitrag hat
  - eine eindeutige ID
  - einen Titel
  - eine Beschreibung
  - das Erstelldatum

```
type Post {
  id: ID!
  title: String!
  description: String!
  created_at: String!
}
```

```
project_root/
|
├── api/
|   ├── __init__.py
|   ├── models.py
|   ├── mutations.py
|   ├── queries.py
|
├── app.py
└── schema.graphql
```

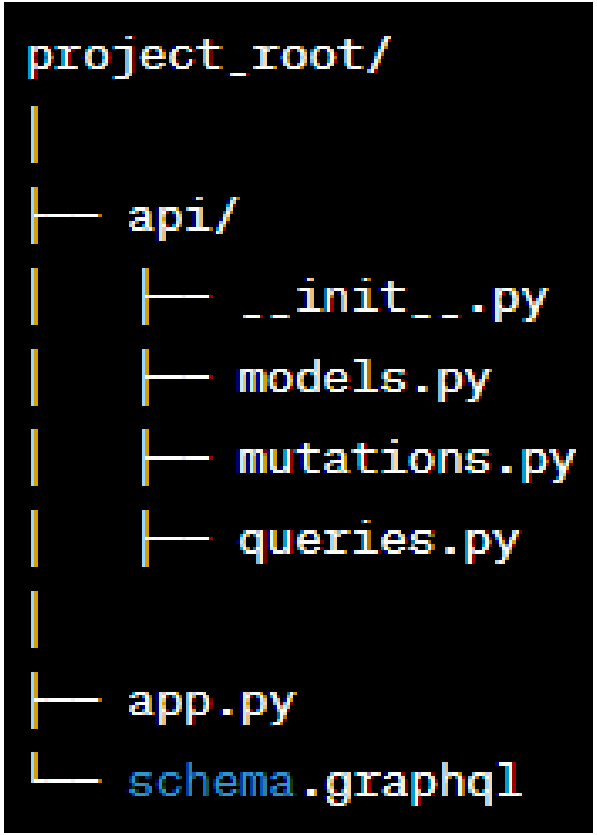
(Haque 2023)

# Tutorial – Python

**schema.graphql**

Keyword: type (Schema mit benutzerdefinierten Objekten (hier Post))

```
type Post {
    id: ID!
    title: String!
    description: String!
    created_at: String!
}
```



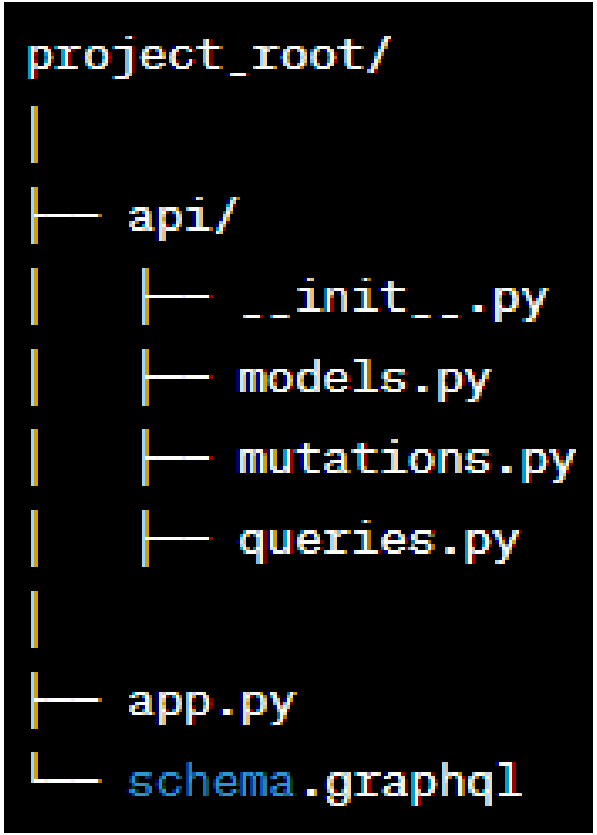

(Haque 2023)

# Tutorial – Python

**schema.graphql**

Jede graphql API hat eine Query, der den entry point für einen Nutzer der API darstellt

```
type Query {
    listPosts: PostsResult!
    getPost(id: ID!): PostResult!
}
```



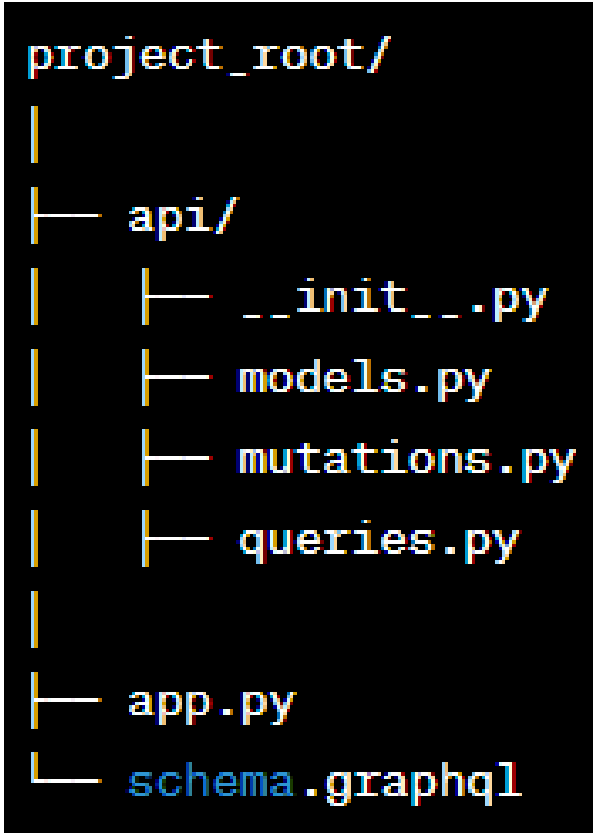

(Haque 2023)

# Tutorial – Python

**schema.graphql**

Wenn Daten auf der API geändert werden sollen → Mutation

```
type Mutation {
    createPost(title: String!, description: String!, created_at: String): PostResult!
}
```

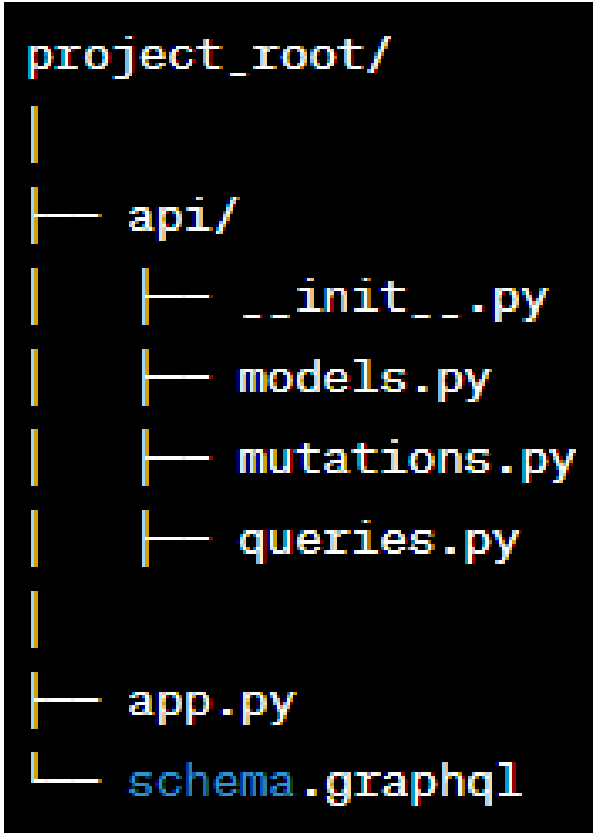



(Haque 2023)

# Tutorial – Python

**Zwischenstand**

- Flask-Server zum Laufen gebracht ✓
- Verbindung zu einer Datenbank hergestellt ✓
- GraphQL-Schema erstellt ✓




(Haque 2023)

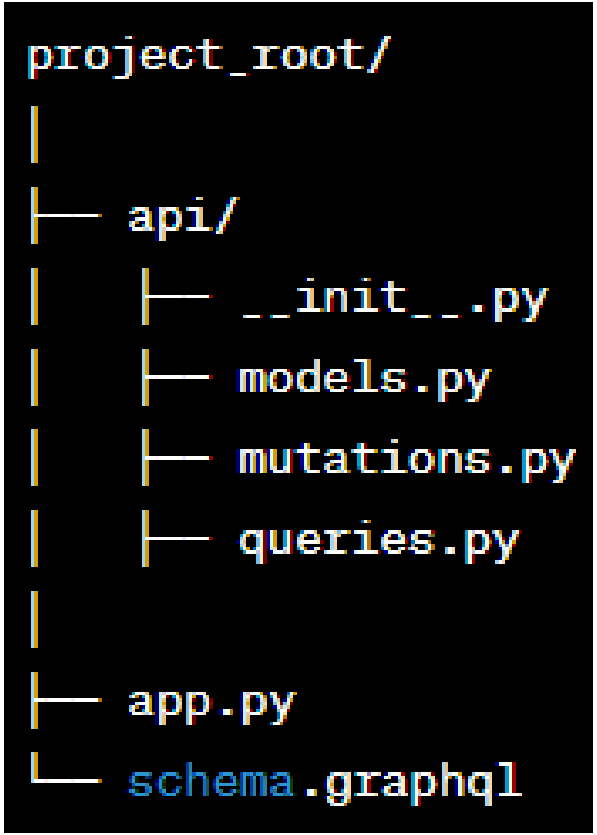
# Tutorial – Python

Flask-Server muss mit GraphQL verbunden werden

➢ Ariadne-Bibliothek

**app.py**

- Funktionen aus der Ariadne-Bibliothek importieren
- Code anpassen




(Haque 2023)

# Tutorial – Apollo Explorer

- kostenlose GraphQL-IDE

- Graph erstellen
➢ Endpoint: http://localhost:5000/graphql

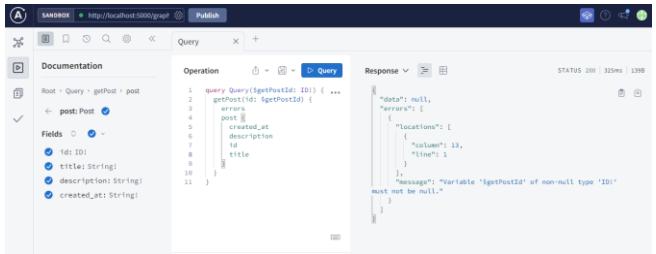
- Flask run
➢ Sollte grün werden


(Apollo Expllorer 2023; Haque 2023)

# Tutorial - Fazit

- Bereitstellung des backend
- jeder Entwickler, der diese API nutzt, kann Abfragen und Datenentitäten erkunden (ohne Programmieren)
- Abfrage wird automatisch vervollständigt, während sie in den Editor eingeben wird

Explorer Query JSON

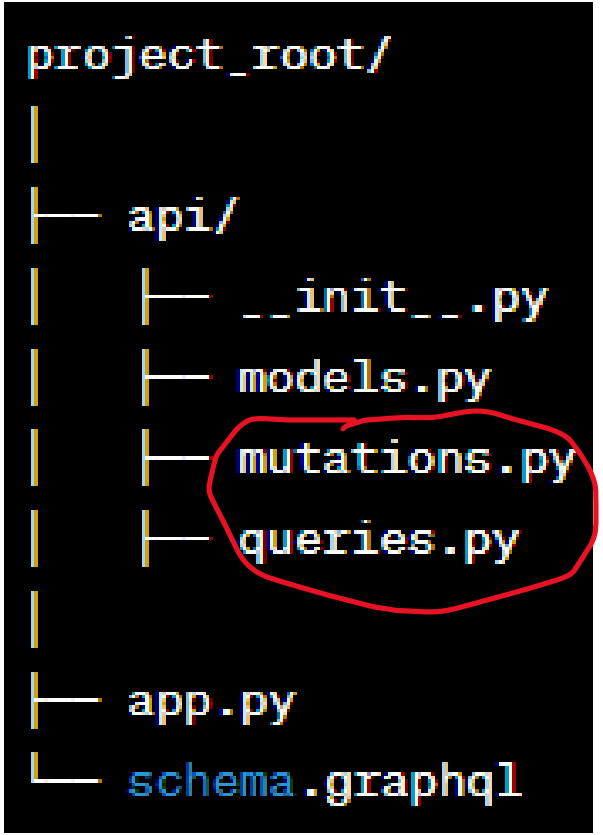


# Tutorial – WAS Fehlt?

**Datenbank befüllen**

**api/queries.py**
- um Abfragen durchzuführen

**api/mutations.py**
- mutation in schema.graphql bringt nichts ohne Mutationsresolver




(Haque 2023)

# Referenzen

Apollo Explorer (2023): The GraphOS Studio Explorer. Online verfügbar unter https://www.apollographql.com/docs/graphos/explorer/, zuletzt aktualisiert am 14.09.2023, zuletzt geprüft am 14.09.2023.

Ariadne (2023): Ariadne · Python GraphQL Schema-first. Online verfügbar unter https://ariadnegraphql.org/, zuletzt aktualisiert am 11.09.2023, zuletzt geprüft am 14.09.2023.

Flask (2023): Welcome to Flask — Flask Documentation (2.3.x). Online verfügbar unter https://flask.palletsprojects.com/en/2.3.x/, zuletzt aktualisiert am 21.06.2023, zuletzt geprüft am 14.09.2023.

GraphQL (2023): GraphQL | A query language for your API. Online verfügbar unter https://graphql.org/, zuletzt aktualisiert am 12.09.2023, zuletzt geprüft am 12.09.2023.

Haque, S. (2023): Using GraphQL with Python – A Complete Guide - Apollo GraphQL Blog. Online verfügbar unter https://www.apollographql.com/blog/graphql/python/complete-api-guide/, zuletzt aktualisiert am 14.09.2023, zuletzt geprüft am 14.09.2023.

FRIEDRICH-SCHILLER-
UNIVERSITÄT
JENA


(GraphQL 2023)

# Vielen Dank für Ihre Aufmerksamkeit!

Natascha Stumpf