Schnittstellen Beschreibung Backend

Die URL fängt immer mit <https://yourChoice.de/api/v1/> an (wird mit … abgekürzt).

Login:

Ein **POST** Request über folgende URL:

…*/login*

Daten:

* hash: … (Der Login Hash Wert)

Logout:

Ein **POST** Request über folgende URL:

…/logout

Daten:

* token: …

Neues Objekt anlegen (speichern):

Hierfür wird ein **POST** request an das *entsprechende* Model geschickt:

*…/<model\_name>*

*…/party*

Daten. Hier werden alle Daten des Objektes [siehe Datenbankschema] übergeben, Beispiel party:

* name: …
* text: …
* consituency: …
* election\_id: …
* token: …

Sollte dies (aus irgendwelchen Gründen) nicht funktionieren, kann es auch über den expliziten Aufruf der save Funktion realisiert werden:

*…/<model\_name>/save*

Objekt bearbeiten:

Hierfür wird ein **PUT** request an das *entsprechende* Model geschickt:

*…/<model\_name>/<id>*

Daten. Hier werden alle Daten des Objektes [siehe Datenbankschema] übergeben, Beispiel party:

* name: …
* text: …
* consituency: …
* election\_id: …
* token: …

Objekt auslesen:

Hierfür wird ein **GET** request an das entsprechende Model geschickt:

*…/<model\_name>/<id>*

Daten:

* token: …

Objekt löschen:

Hierfür wird ein **DELETE** request an das entsprechende Model geschickt:

*…/<model\_name>/<id>*

Daten:

* token: …

Wahl auswerten:

Ein **POST** Request über folgende URL:

*…/election/evaluate*

Daten:

* id: …
* token: …

Wählen:

Ein **POST** Request über folgende URL:

…/vote/

Daten:

* election\_id: …
* first\_vote: …
* second\_vote: …
* valid: …

Oder ein **POST** Request über folgende URL:

…/election/vote

Daten:

* election\_id: …
* first\_vote: …
* second\_vote: …
* valid: …

*Es besteht die Möglichkeit, dass die jeweilige ID als Parameter, anstatt als URI mitgeschickt werden muss*