# Erläuterung des Datenmodells

Das Datenmodell für die Quiz-App besteht aus fünf selbst erstellten Entitäten. Daneben erstellt das Django-Framework eigene Entitäten, die für das User-Management benötigt werden. IVm den selbst erstellten Entitäten ist davon nur die „auth\_user“ Entität unmittelbar relevant.

## Erläuterung der Entitäten

Die Folgenden Modelle werden für die Bibliothek bzw. dem Management der Quiztasks benutzt. Sie werden in der Datei „Library\models.py“ definiert.

### QuizPool

Quizpools dienen dazu Quiztasks zu einem bestimmten Thema zu bündeln.

Attribute:

* name: Ein Zeichenfeld, das den Namen des Quiz-Pools speichert.
* creator: Ein Fremdschlüssel-Feld, das auf das Benutzermodell verweist und den Benutzer darstellt, der den Quiz-Pool erstellt hat.
* created\_date und updated\_date: Datumfelder, die automatisch den Zeitstempel setzen, wann das Modell erstellt bzw. aktualisiert wurde.

### QuizTask

Kern der Quiztasks ist eine Frage, die einem Anwender im Rahmen des Quiz gestellt wird. Einem Quiztask mehrere mögliche Antworten zugeordnet.

Attribute:

* pool: Ein Fremdschlüssel-Feld, das auf das QuizPool-Modell verweist und den Quiz-Pool darstellt, zu dem diese Aufgabe gehört.
* question: Ein Zeichenfeld, das die Frage enthält.
* approved: Ein Boolesches Feld, das angibt, ob die Frage von einem Dozenten genehmigt ist.
* creator: Ein Fremdschlüssel-Feld, das auf das Benutzermodell verweist und den Benutzer darstellt, der die Quiz-Aufgabe erstellt hat.

### Answer

Im Answer-Model werden mögliche Antworten zu einer Quiztask gespeichert. Jede Antwort kann als richtig bzw. falsch markiert werden. Für jede Antwort kann eine Begründung für die richtige bzw. falsche Lösung hinterlegt werden.

Attribute:

* task: Ein Fremdschlüssel-Feld, das auf das QuizTask-Modell verweist und die Quiz-Aufgabe darstellt, zu der diese Antwort gehört.
* answer: Ein Zeichenfeld, das die Antwort enthält.
* correct: Ein Boolesches Feld, das angibt, ob die Antwort korrekt ist oder nicht.
* explanation: Ein Zeichenfeld, das die Erklärung für die Antwort enthält.

Die Folgenden Modelle sind werden für das Quiz im Einzelspielermodus genutzt. Sie werden in der Datei „Singleplayer\models.py“ definiert.

### SPGame

Im Singeplayer-Game (SPGame) werden Einzelspieler-Objekte gespeichert.

Attribute:

* user: Ein Fremdschlüssel-Feld, das auf das Benutzermodell verweist und den Benutzer darstellt, der mit dem Spiel verbunden ist.
* pool: Ein Fremdschlüssel-Feld, das auf das QuizPool-Modell verweist und den Quiz-Pool darstellt, der mit dem Spiel verbunden ist.
* created\_date: Datumfeld, das automatisch den Zeitstempel setzt, wann das Modell erstellt wurde.
* name: Ein Zeichenfeld, das den Namen des Spiels speichert.
* completed: Ein Boolesches Feld, das anzeigt, ob das Spiel abgeschlossen ist oder nicht.
* completed\_at: Ein Datumsfeld, das den Zeitstempel speichert, wann das Spiel abgeschlossen wurde.
* correct\_percent: Ein Float-Feld, das den Prozentsatz korrekter Antworten speichert.

### SPGame\_contains\_Quiztask

Jedem SPGame werden über das SPGame\_contains\_Quiztask Modell Quiztasks entsprechend des auswegählten Quizpools zugeordnet. SPGame\_contains\_Quiztask entspricht dabei der Umsetzung einer „m-zu-n“-Beziehung. Neben der Herstellung der Beziehung zwischen Quiztask und SPGame werden noch weitere Spiel-Spezifische Eigenschaften gespeichert.

Attribute:

* game: Ein Fremdschlüssel-Feld, das auf das SPGame-Modell verweist und das Spiel darstellt, das mit der Quiz-Aufgabe verbunden ist.
* task: Ein Fremdschlüssel-Feld, das auf das QuizTask-Modell verweist und die Quiz-Aufgabe darstellt, die mit dem Spiel verbunden ist.
* selected\_answer: Ein Fremdschlüssel, der auf das Answer-Modell verweist und die vom Benutzer für die Quiz-Aufgabe ausgewählte Antwort darstellt.
* correct\_answered: Ein Boolesches Feld, das angibt, ob die Quiz-Aufgabe korrekt beantwortet wurde oder nicht.
* completed: Ein Boolesches Feld, das angibt, ob die Aufgabe abgeschlossen ist oder nicht.