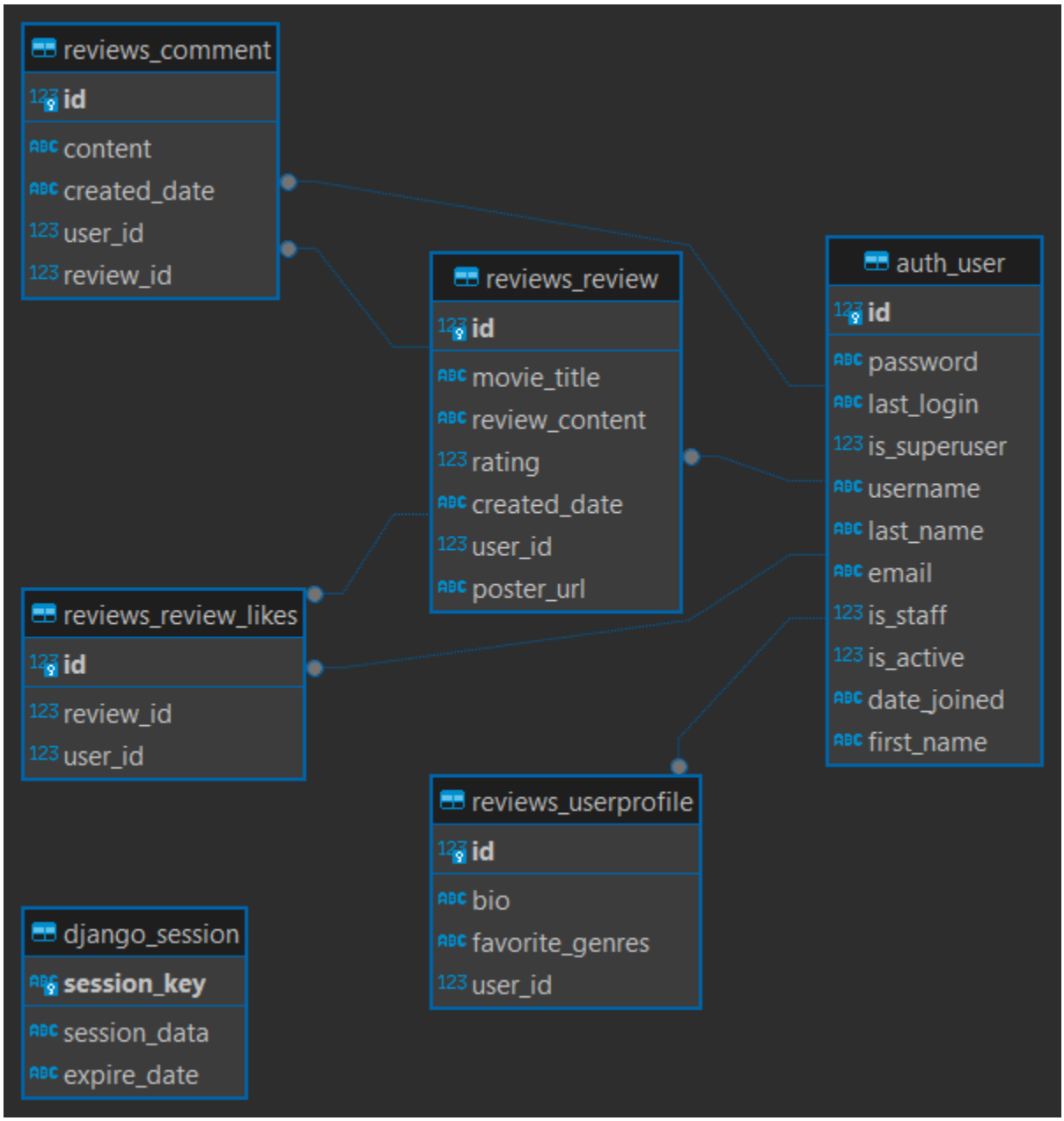
API Database Design: Movie Review API

Entity-Relationship Diagram (ERD)



Deployed Web APP Link: Movie Review API

https://hashimaziz88.pythonanywhere.com/

View the Slide Deck: Movie_Review_API Slides Hashim

https://docs.google.com/presentation/d/1bTdwqfRYv3OmdvDa6qfXhc6zR07JWXpJhn3UcswmrPA/edit?usp=sharing

Entities and Relationships:

auth_user

- id (PK)
- password
- last_login
- is_superuser
- username
- first_name
- last_name
- email
- is_staff
- is_active
- date_joined

reviews_userprofile

- id (PK)
- bio
- favorite_genre
- user_id (FK to auth_user (id), One-to-One)

reviews_review

- id (PK)
- movie_title
- review_content
- rating_value
- created_date

- user_id (FK to auth_user (id))

reviews_comment

- id (PK)
- content
- created_date
- user_id (FK to auth_user (id))
- review_id (FK to reviews_review (id))

Relationships:

➔ auth_user has a one-to-many relationship with reviews_review.
➔ auth_user has a one-to-one relationship with reviews_userprofile.
➔ reviews_review has a one-to-many relationship with reviews_comment.
➔ auth_user has a many-to-many relationship with reviews_comment (if users can comment on multiple reviews and reviews can have comments from multiple users).

Additional Notes:

- The auth_user entity uses Django's built-in User model.
- The reviews_review entity connects users to their movie reviews, storing both the review content and rating.
- The reviews_userprofile entity stores additional user information without modifying the core User model.
- The reviews_comment entity allows users to comment on reviews, linking comments to both users and reviews.