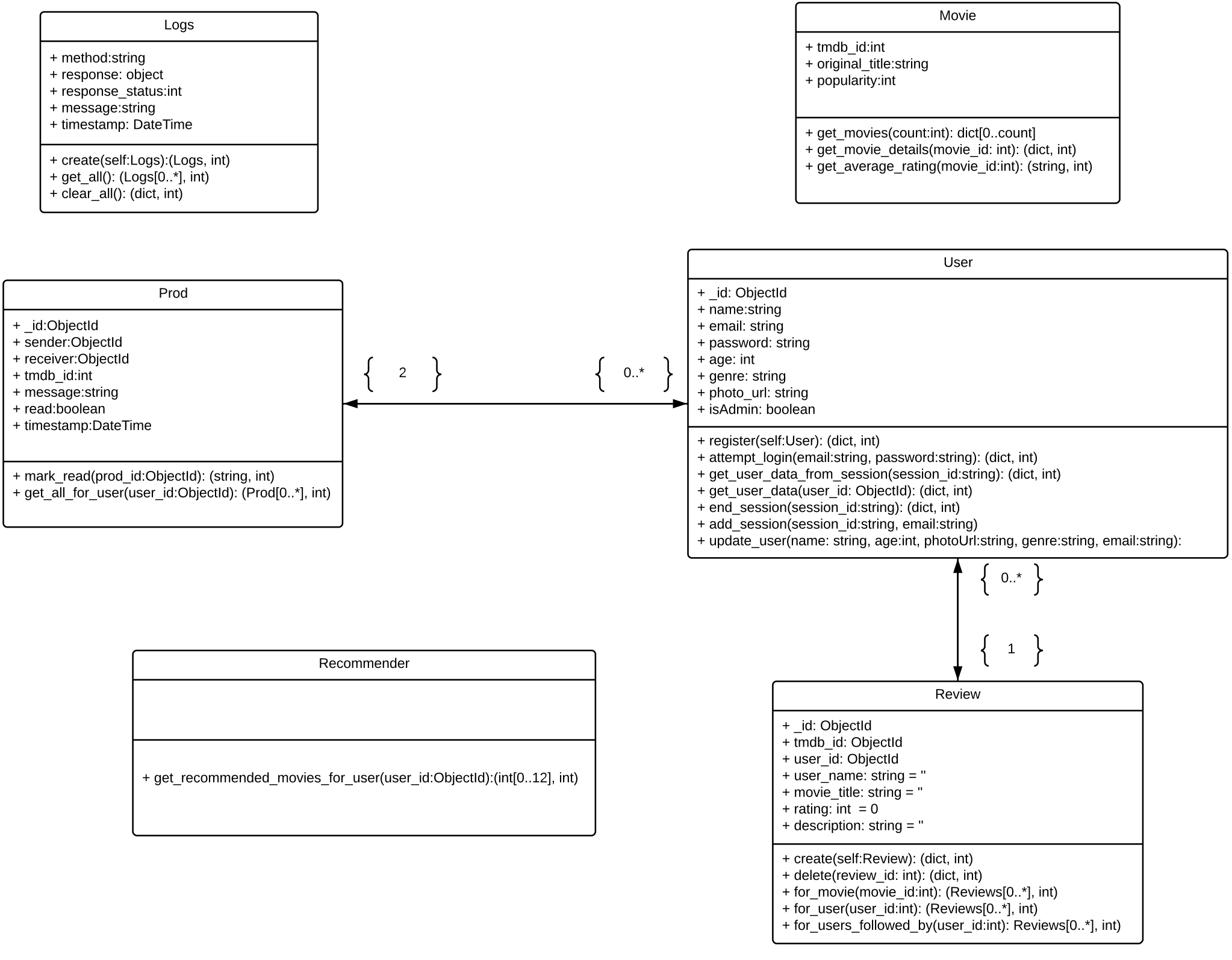
Logs
+ method:string
+ response: object
+ response_status:int
+ message:string
+ timestamp: DateTime
+ create(self:Logs):(Logs, int)
+ get_all(): (Logs[0..*], int)
+ clear_all(): (dict, int)
Movie
+ tmdb_id:int
+ original_title:string
+ popularity:int
+ get_movies(count:int): dict[0..count]
+ get_movie_details(movie_id: int): (dict, int)
+ get_average_rating(movie_id:int): (string, int)
Prod
+ _id:ObjectId
+ sender:ObjectId
+ receiver:ObjectId
+ tmdb_id:int
+ message:string
+ read:boolean
+ timestamp:DateTime
+ mark_read(prod_id:ObjectId): (string, int)
+ get_all_for_user(user_id:ObjectId): (Prod[0..*], int)
2
0..*
User
+ _id: ObjectId
+ name:string
+ email: string
+ password: string
+ age: int
+ genre: string
+ photo_url: string
+ isAdmin: boolean
+ register(self:User): (dict, int)
+ attempt_login(email:string, password:string): (dict, int)
+ get_user_data_from_session(session_id:string): (dict, int)
+ get_user_data(user_id: ObjectId): (dict, int)
+ end_session(session_id:string): (dict, int)
+ add_session(session_id:string, email:string)
+ update_user(name: string, age:int, photoUrl:string, genre:string, email:string):
0..*
1
Recommender
+ get_recommended_movies_for_user(user_id:ObjectId):(int[0..12], int)
Review
+ _id: ObjectId
+ tmdb_id: ObjectId
+ user_id: ObjectId
+ user_name: string = ''
+ movie_title: string = ''
+ rating: int = 0
+ description: string = ''
+ create(self:Review): (dict, int)
+ delete(review_id: int): (dict, int)
+ for_movie(movie_id:int): (Reviews[0..*], int)
+ for_user(user_id:int): (Reviews[0..*], int)
+ for_users_followed_by(user_id:int): Reviews[0..*], int)