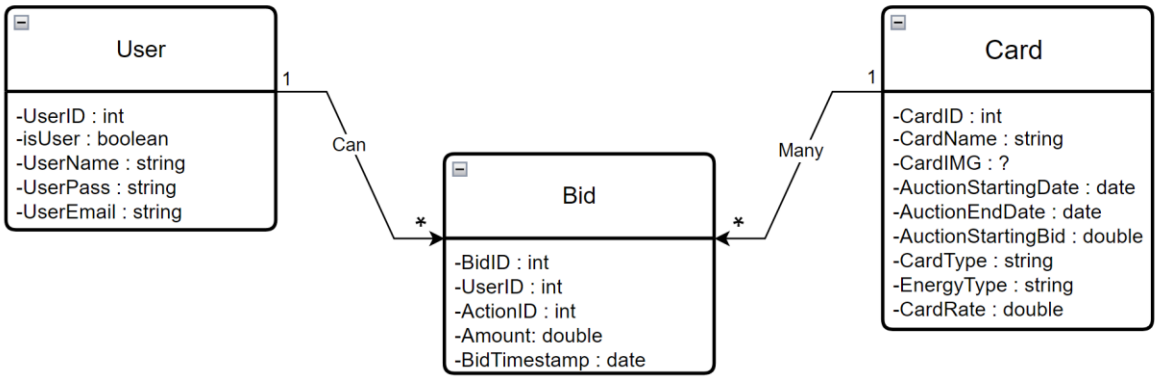
Web Advanced API Documentation

Table of Content

Class diagram

This is my class diagram with all my 3 entities (Users, Bids and Cards)

A User can place multiple Bids (1..*), but each Bid is associated with only one User (1).
This relationship ensures that bids are assign to a specific user.
An Card can have multiple Bids (1..*), but each Bid is associated with only one Card (1).
This relationship ensures that bids are connected to a specific Card.



GET requests

GET	http://localhost:3000/users/cards		
Gets all the won cards from the user			
Parameters:	**Name**	**Type**	**Description**
** required*			
Responses:	**Code**	**Description / example if successful**	
	200	Return a list of all won cards from the user	

GET	http://localhost:3000/cards		
Retrieve all Cards registered on the website			
Parameters:	**Name**	**Type**	**Description**
** required*		path	Display all cards
	cardName	querry	Filter cards by name
	cardType	querry	Filter cards by type
	energyType	querry	Filter cards by energy
	cardRate	querry	Filter cards by rate
Responses:	**Code**	**Description / example if successful**	
	200	Return a list of all Cards	

GET	http://localhost:3000/cards/popular		
Retrieve the popular cards of the site (random 5 cards)			
Parameters:	**Name**	**Type**	**Description**
** required*			Retrieve the popular cards of the site
Responses:	**Code**	**Description / example if successful**	
	200	Returns the User with the ID that was provided.	
	404	User ID is incorrect	

GET	http://localhost:3000/cards/{cardID}		
Retrieve the Card by ID			
Parameters:	**Name**	**Type**	**Description**
** required*	cardID	path	Id of the card to get the information
Responses:	**Code**	**Description / example if successful**	
	200	Returns the Card Information with the ID that was provided.	
	404	Card ID is incorrect	

GET	http://localhost:3000/cards/{cardD}/bids		
Retrieve all bids from the selected Card by Its ID			
Parameters:	**Name**	**Type**	**Description**
** required*	cardID	path	Id of the card to get bids
Responses:	**Code**	**Description / example if successful**	
	200	Return a list of all the bids from the selected Card	
	404	Card ID not found	

POST requests

POST	http://localhost:3000/tokens/		
User will login with his credentials and will create a Token			
Parameters:	**Name**	**Type**	**Description**
** required*	User	body	The User information needed to be in a JSON format to login (email,passord)
Responses:	**Code**	**Description / example if successful**	
	200	User Login successfully, token created and given to user	
	404	Invalid input data - Email or Password not found	

POST	http://localhost:3000/users		
A new User will be created/registred			
Parameters:	**Name**	**Type**	**Description**

** required*	User	body	The user data to be added. Must be JSON with email and password.
			
			Example: {
			"email": "Rt01@gmail.com",
			"password": "Rt14"
			}

Responses:	**Code**	**Description / example if successful**	
	200	A new User will be created	
	404	Invalid input data - Email or Password not found	
	500	Internal server error with the error message.	

POST	http://localhost:3000/bids/{cardID}		
A new Bid will be to the Card			
Parameters:	**Name**	**Type**	**Description**
** required*	Card ID	path	Needs the Card ID to be added

	Bid	body	The user data to be added. Must be JSON with email and password.
			
			Example: {
			"bidAmount": 22277
			}

Responses:	**Code**	**Description / example if successful**	
	201	A new Bid will Is created	
	400	Invalid input data	
	404	Card not found	

POST	http://localhost:3000/cards		
A new Card will be added			
Parameters:	**Name**	**Type**	**Description**

** required*	Card	body	The user data to be added. Must be JSON with email and password.
			
			Example: {
			"cardName": "Alakazam",
			"cardImg": "aaa.png",
			"actionStartingDate": "14/12/2024",
			"auctionEndDate": "14/01/2025"
			"auctionStartingBid": 150.00,
			"cardType": "Pokemon",
			"energyType": "Psychic",
			"cardRate": 7.5,
			}

Responses:	**Code**	**Description / example if successful**	
	201	A new Card will be added	
	404	Invalid input data	

PATCH requests

PUT	http://localhost:3000/cards/{cardID}		
Update the selected Card by its ID			
Parameters:	**Name**	**Type**	**Description**
** required*	CardID	path	Needs the Card ID to be edit

	Card	body	The user data to be added. Must be JSON with email and password.
			
			Example: {
			"cardName": "Alakazam",
			"cardImg": "aaa.png",
			"actionStartingDate": "14/12/2024",
			"auctionEndDate": "14/01/2025"
			"auctionStartingBid": 150.00,
			"cardType": "Pokemon",
			"energyType": "Psychic",
			"cardRate": 7.5,
			}

Responses:	**Code**	**Description / example if successful**	
	200	Card updated successfully	
	400	All errors 404 will be town here.	
	404	Card not found	

DELETE requests

DELETE	**http://localhost:3000/cards/{cardID}**		
Deletes the Card by his ID			
Parameters:	**Name**	**Type**	**Description**
** required*	cardID	path	Needs the Card ID to be deleted
Responses:	**Code**	**Description / example if successful**	
	204	Card deleted successfully	
	404	Card not found	