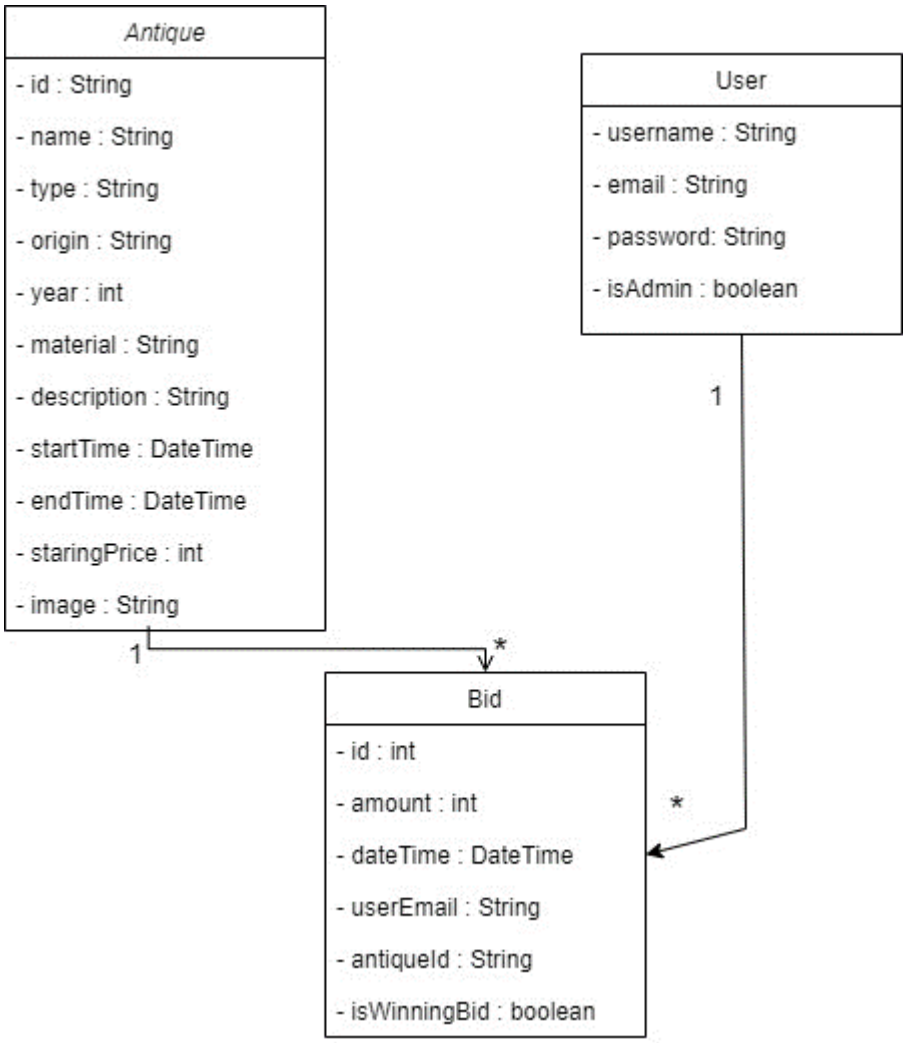
API SPECIFICATION

DHI2V.So
DIMITAR PEVETSOV
WEB ADVANCED

Table of Contents

1. Class diagram



2. GET requests

GET	**/antiques**		
Gets all the antiques			
Parameters:	**Name**	**Type**	**Description**
** required*	year	query	Returns antiques based on the given year
	Origin	query	Return antiques based on the given origin
	Material	query	Returns antiques based on the given material
	Search	query	Returns antiques based on the given search prompt
Responses:	**Code**	**Description / example if successful**	
	200	Got the list of antiques which can be empty	

GET	**/antiques/{id}**		
Gets antique with the specified id			
Parameters:	**Name**	**Type**	**Description**
** required*	Id*	path	Id of the antique to find
Responses:	**Code**	**Description / example if successful**	
	200	Got the data for the antique with the specified id	
	404	An antique with that id is not found	

GET	**/users**		
Get all users			
Parameters:	**Name**	**Type**	**Description**
** required*			
Responses:	**Code**	**Description / example if successful**	
	200	Got the list of users which can be empty	

GET	**/users/{email}**
Gets the user with the specified email	

Parameters:	Name	Type	Description
* *required*	email*	path	Email of user to find

Responses:	Code	Description / example if successful
	200	Got the data for the user with the specified email
	404	A user with that email is not found

GET	/antiques/{id}/bids
Gets all bids for the antique with the specified id	

Parameters:	Name	Type	Description
	Id*	path	Id of antique to find
* *required*			

Responses:	Code	Description / example if successful
	200	Got the list of bids which can be empty
	404	No bids found for that antique

GET	/users/{email}/bids
Gets the bids for the user with the specified email	

Parameters:	Name	Type	Description
* *required*	Email*	path	Email of user to find

Responses:	Code	Description / example if successful
	200	Got the list of bids which can be empty
	404	No bids found for that user

GET	/users/{email}/winners
Get the winning bids for the user with the specified email	

Parameters:	Name	Type	Description
	Email*	Path	Email of user to find

Responses:	Code	Description / example if successful
	200	Got the list of winning bids which can be empty

3. POST requests

POST	/antiques		
Adds an antique			
Parameters:	**Name**	**Type**	**Description**
** required*	antique*	body	The antique being added. JSON body gets filled
Responses:	**Code**	**Description / example if successful**	
	201	Antique created successfully and JSON body gets filled	
	400	Antique could not be created, because fields are empty or invalid	

POST	/users		
Adds a user			
Parameters:	**Name**	**Type**	**Description**
** required*	user*	body	The user that is being added. JSON body gets filled
Responses:	**Code**	**Description / example if successful**	
	201	User created successfully and JSON body gets filled	
	400	User could not be created, because fields are empty or invalid	
	500	User could not be created, because there is an error with the hashing of password or JWT creation.	

POST	/antiques/{id}/bids		
Creares a bid for an antique with the specified id			
Parameters:	**Name**	**Type**	**Description**
** required*	bid*	body	The bid that is being created. JSON body gets gilled.
	Id*	Path	Id of antique to find
Responses:	**Code**	**Description / example if successful**	
	201	Bid created successfully and JSON body gets filled	
	400	Bid could not be created, because fields are empty or invalid	

POST	/tokens
Creating a token for logging in.	

Parameters:	**Name**	**Type**	**Description**
	Token*	Body	The token that is being created

Responses:	**Code**	**Description / example if successful**
	201	Token successfully created.
	400	Token was not created as the email and password in the body was not found or invalid
	500	Token was not created as there was an error creating the JWT or comparing the password from the body.

4. PATCH requests

PATCH	**antiques/{id}**		
Modifies an antique, based on the given id			
Parameters:	**Name**	**Type**	**Description**
** required*	id *	path	The id of the antique to be modified
	antique *	body	The antique body to modify
Responses:	**Code**	**Description / example if successful**	
	200	Body contains the modified antique.	
	400	Body was not modified, because a field is invalid or empty when trying to modify or body id does not match the path id	

5. DELETE requests

DELETE	**/antiques/{id}**		
Deletes the antique with the specified id			
Parameters:	**Name**	**Type**	**Description**
** required*	id*	path	Id of the antique to delete
Responses:	**Code**	**Description / example if successful**	
	200	Antique deleted	
	404	A antique with that Id is not found	

DELETE	**/antiques/{id}/bids/{id}**		
Deletes data for bid with the specified id			
Parameters:	**Name**	**Type**	**Description**
** required*	id*	path	Id of the bid to delete
	Id*	path	Id of the antique the bid belongs to
Responses:	**Code**	**Description / example if successful**	
	200	Bid deleted	
	404	A bid or antique with that Id is not found	
	400	Bid was not deleted as auction has ended	