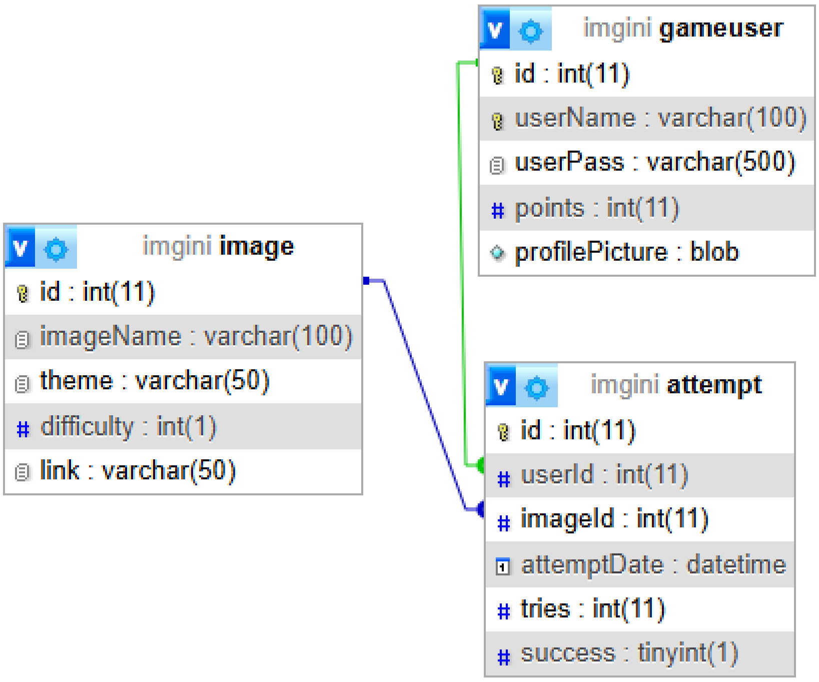
Imgini DB & API documentation

Database Structure



imgini **gameuser**
id : int(11)
userName : varchar(100)
userPass : varchar(500)
points : int(11)
profilePicture : blob

imgini **image**
id : int(11)
imageName : varchar(100)
theme : varchar(50)
difficulty : int(1)
link : varchar(50)

imgini **attempt**
id : int(11)
userId : int(11)
imageId : int(11)
attemptDate : datetime
tries : int(11)
success : tinyint(1)

The ImGini database consists of three related tables:

- **Users**: Stores registered user data, including ID, name, password, streak score, and profile picture.
- **Images**: Contains all images used in the game, with their respective ID, name, theme, difficulty, and image link.

- **Attempts**: Records daily player attempts, storing attempt ID, image and user IDs, attempt date, number of attempts, and whether it was successful.

API Endpoints

1. View Daily Image

- **Endpoint:** `GET /imgini/dailyImage`
- **Parameters:**

Parameter	Type	Description
userToken	String	User authentication token

- **Response:** JSON with daily image data.

2. View Random Image

- **Endpoint:** `GET /imgini/infiniteImage`
- **Parameters:**

Parameter	Type	Description
userToken	String	User authentication token
imgTheme	String	Image theme

- **Response:** JSON with a random image's data.

3. Logout

- **Endpoint:** `GET /imgini/logout`
- **Parameters:**

Parameter	Type	Description
userToken	String	User authentication token

- **Response:**
  - HTTP 204 (NO_CONTENT) if logout was successful.
  - HTTP 404 (NOT_FOUND) if the user does not exist.

4. Get Player Ranking

- **Endpoint:** `GET /imgini/ranking`
- **Parameters:**

Parameter	**Type**	**Description**
userToken	String	User authentication token
page	Int	Page number
size	Int	Number of users per page

- **Response:** JSON with player ranking data.

5. Get User Information

- **Endpoint:** `GET /imgini/userInfo`
- **Parameters:**

Parameter	**Type**	**Description**
userToken	String	User authentication token
username	String	Username to query

- **Response:** JSON with user information.

6. Streak

- **Endpoint:** `GET /imgini/streak`
- **Parameters:**

Parameter	**Type**	**Description**
userToken	String	User authentication token
username	String	Username
page	Int	Page number
size	Int	Number of attempts per page

- **Response:** JSON with attempt history and current streak.

7. ImgThemes

- **Endpoint:** `GET /imgini/imgsThemes`
- **Parameters:**

Parameter	Type	Description
userToken	String	User authentication token

- **Response:** Array with all available image themes.

8. Get Today's Attempt

- **Endpoint:** `GET /imgini/getAttempt`
- **Parameters:**

Parameter	Type	Description
userToken	String	User authentication token
username	String	Username

- **Response:** JSON with today's attempt data.

9. User Registration

- **Endpoint:** `POST /imgini/register`
- **Parameters:**

Parameter	Type	Description
userDTO	Object	Object with username and password

- **Response:**
  - HTTP 403 (FORBIDDEN) if the user already exists.
  - HTTP 200 (OK) if successfully registered.

10. Login

- **Endpoint:** `POST /imgini/login`
- **Parameters:**

Parameter	Type	Description
userDTO	Object	Object with username and password

- **Response:**
  - HTTP 404 (NOT_FOUND) if the user does not exist.
  - HTTP 200 (OK) if credentials match.

11. Register Attempt

- **Endpoint:** `POST /imgini/newAttempt`
- **Parameters:**

Parameter	Type	Description
attemptDTO	Object	Object with attempt data
userToken	String	User authentication token

- **Response:**
  - HTTP 403 (FORBIDDEN) if an attempt has already been made today.
  - HTTP 204 (NO_CONTENT) if the attempt was successfully registered.

12. Delete User

- **Endpoint:** `DELETE /imgini/delete`
- **Parameters:**

Parameter	Type	Description
name	String	Username
password	String	User password
userToken	String	User authentication token

- **Response:**
  - HTTP 404 (NOT_FOUND) if the user does not exist.

- HTTP 204 (NO_CONTENT) if the user was deleted.

13. Update User

- **Endpoint:** `PUT /imgini/update`
- **Parameters:**

Parameter	**Type**	**Description**
userToken	String	User authentication token
updatedUser	Object	Object with updated user data

- **Response:**
  - HTTP 404 (NOT_FOUND) if the user does not exist.
  - HTTP 204 (NO_CONTENT) if successfully updated.