Aggie's House

Aggie's House Team

May 01, 2024

CONTENTS:

CHAPTER

ONE

CHAT ROUTES

Endpoints used for the chat window.

routes.chat.routes.**get_user**()

Endpoint to retrieve information about the current authenticated user.

Returns:

Response: a JSON response containing user data for the current user. - If the user is not authenticated, returns a JSON response with an error message and a status code 401. - If the user is not found in the database, returns a JSON response with an error message and a status code 404.

CHAPTER

TWO

FEED ROUTES

Endpoints used for the feed.

routes.feed.routes.**calculate_age**(*birthday*)

Calculate age based on the provided birthday.

Parameters:

- birthday (date): The date of birth of the user.

Returns:

int or None: The calculated age of the user, or None if the birthday is not provided.

routes.feed.routes.**check_for_match**(*user_id_1*, *user_id_2*)

Check if two users have mutually liked each other.

Parameters:

- user_id_1 (int): The ID of the first user.
- user_id_2 (int): The ID of the second user.

Returns:

bool: True if both users have liked each other, False otherwise.

routes.feed.routes.**compare_users_to_current**(*current_user*, *users*)

Compare current user to other users based on various attributes.

Parameters:

- current_user (User): The current user for comparison.
- users (list): List of User objects to compare against.
- age_similarity_weight (float): Weight for age similarity comparison.
- bio_similarity_weight (float): Weight for biography similarity comparison.
- major_similarity_weight (float): Weight for major similarity comparison.
- gender_similarity_weight (float): Weight for gender similarity comparison.
- answer_similarity_weight (float): Weight for answer similarity comparison.

Returns:

list: A list of tuples containing user IDs and their overall similarity scores compared to the current user.

routes.feed.routes.**dislike**()

Endpoint to dislike another user.

Parameters:

liked_user_id (int): The ID of the user being disliked.

Returns:

Response: A JSON response indicating that the dislike was successful.

- message (str): A message indicating the result of the dislike operation.

Raises:
ValueError: If the request does not contain JSON data. ValueError: If either the user ID or the liked user ID is missing from the request.

routes.feed.routes.**getfeed**()

Endpoint to fetch the feed of potential matches for the current user.

Returns:
Response: A JSON response containing a list of user IDs representing potential matches.

Raises:
ValueError: If the current user is not authenticated.

routes.feed.routes.**like**()

Endpoint to like another user.

Parameters:
liked_user_id (int): The ID of the user being liked.

Returns:

Response: A JSON response indicating whether the like was successful and if it resulted in a match.

- message (str): A message indicating the result of the like operation.
- match (bool): Indicates whether the like resulted in a match.

Raises:
ValueError: If the request does not contain JSON data. ValueError: If either the user ID or the liked user ID is missing from the request.

routes.feed.routes.**unmatch**(*matched_user_id*)

Endpoint to unmatch with a previously matched user.

Parameters:
matched_user_id (int): The ID of the user to unmatch.

Returns:

Response: A JSON response indicating whether the unmatch was successful.

- message (str): A message indicating the result of the unmatch operation.

Raises:
ValueError: If the method is not POST. ValueError: If no matching entry is found for the current user and the matched user.

routes.feed.routes.**userinfo**()

Endpoint to retrieve user information.

Parameters:
user_id (int): The ID of the user whose information is requested.

Returns:

Response: A JSON response containing the user's information.

- first_name (str): The user's first name.

- age (int): The user's age.
- major (str): The user's major.
- bio (str): The user's biography.
- categories (dict): A dictionary containing categories and their corresponding answered questions.
- photos (list): A list of URLs to the user's photos.
- gender (str): The user's gender.
- verified (bool): Whether the user is verified.
- email (str): The user's email.

Raises:

ValueError: If no user ID is provided or if the user is not found.

CHAPTER

THREE

MATCHES ROUTES

Endpoints used for the matches page

routes.matches.routes.**getmatches**()

Get matches for the current user.

Returns:

Response: A JSON response containing information about the matches.

CHAPTER

FOUR

IMAGES ROUTES

Endpoints used for image display and submission.

routes.images.routes.**allowed_file**(*filename*)

Check if a file has an allowed extension.

Parameters:

- filename (str): The name of the file to check.

Returns:

bool: True if the file has an allowed extension, False otherwise.

routes.images.routes.**extract_filename_from_presigned_url**(*presigned_url*)

Extract the filename from a presigned URL.

Parameters:

- presigned_url (str): The presigned URL from which to extract the filename.

Returns:

str: The extracted filename.

routes.images.routes.**get_profile_images**()

Retrieve profile images associated with the current user from AWS bucket.

Returns:

Response: A JSON response containing the URLs of the user's profile photos.

routes.images.routes.**save_profile_images**()

Save profile images associated with the current user.

Returns:

Response: A JSON response indicating whether the files were successfully uploaded.

routes.images.routes.**upload_photo_aws**(*user*, *image*, *index*)

Upload a photo to AWS S3 or set it as the user's profile photo.

Parameters:

- user (User): The user for whom the photo is being uploaded.
- image (str or FileStorage): The image to upload.
- index (int): The index of the image in the array of photos.

Returns:

None

routes.images.routes.**upload_photo_local**(*user*, *image*, *index*)

Upload a photo locally or set it as the user's profile photo.

Parameters:

- user (User): The user for whom the photo is being uploaded.
- image (str or FileStorage): The image to upload.
- index (int): The index of the image in the array of photos.

Returns:

None

CHAPTER

FIVE

LOGIN ROUTES

Endpoints used for login.

routes.login.routes.**check_profile_status**()

Check the profile completion status of the current user.

Returns:
Response: A JSON response containing the profile completion status.

routes.login.routes.**gauthorize**()

Authorize a Google login request.

Returns:
Response: A JSON response indicating the success or failure of the login attempt.

routes.login.routes.**get_majors**()

Get a list of available majors.

Returns:
Response: A JSON response containing the list of majors.

routes.login.routes.**login**()

Log in a user with email and password.

Returns:
Response: A JSON response indicating the success or failure of the login attempt.

routes.login.routes.**logout_user**()

Log out the current user.

Returns:
Response: A JSON response indicating the success or failure of the logout attempt.

routes.login.routes.**signup**()

Register a new user.

Returns:
Response: A JSON response indicating the success or failure of the sign-up attempt.

routes.login.routes.**verify_google_token**(*access_token*)

Verify a Google token using Authlib.

Parameters:
access_token (str): The access token to verify.

Returns:

dict or None: A dictionary containing user information if verification is successful,
None otherwise.

CHAPTER

SIX

QUESTIONNAIRE ROUTES

Endpoints used for the questionnaire.

routes.questionnaire.routes.**get_profile_info**()

Endpoint to retrieve profile information of the logged-in user.

Returns:
Response: A JSON response containing the user's profile information.

routes.questionnaire.routes.**get_questions_by_category**(*category_id*)

Get questions by category ID.

Args:
category_id (int): The ID of the category.

Returns:
Response: A JSON response containing questions and their answer choices.

routes.questionnaire.routes.**get_user_answers_by_category**(*category_id*)

Endpoint to retrieve user answers by category ID.

Args:
category_id (int): The ID of the category.

Returns:
Response: A JSON response containing user answers for questions in the specified category.

routes.questionnaire.routes.**get_user_id**()

Endpoint to retrieve the user ID from the session.

Returns:
Response: A JSON response containing the user ID if found, or an error message if not found.

routes.questionnaire.routes.**save_answers**()

Endpoint to save user answers to questions.

Returns:
Response: A JSON response indicating the success of saving the answers.

routes.questionnaire.routes.**update_profile_info**()

Endpoint to update the profile information of the logged-in user.

Returns:
Response: A JSON response indicating the success or failure of the profile update.

CHAPTER

SEVEN

SETTINGS ROUTES

Endpoints used for the edit profile page.

routes.settings.routes.**change_active_status**()

Endpoint to toggle the active status of the current user (deactivate/reactivate a user).

Returns:
Response: A JSON object confirming the successful status change.

routes.settings.routes.**check_active_status**()

Endpoint to check the active status of the current user (whether the user is deactivated or not).

Returns:
Response: A JSON object indicating whether the user is active or not.

routes.settings.routes.**delete_account**()

Endpoint to delete the current user's account.

Returns:
Response: A JSON object indicating the success of the deletion process.

routes.settings.routes.**delete_categories**()

Endpoint to delete all answers associated with selected categories for the current user.

Args:
categories (list): A list of category IDs to delete answers from.

Returns:
Response: A JSON object confirming the deletion of answers.

routes.settings.routes.**edit_answers**()

Endpoint to edit user answers for questions.

Args:
request.json (dict): A JSON object containing the updated answers keyed by question IDs.

Returns:
Response: A JSON response indicating the success or failure of the operation.

routes.settings.routes.**get_category_id**()

Endpoint to retrieve the category ID based on the provided category name.

Returns:
Response: A JSON response containing the category ID if found, or an error message if not found.

routes.settings.routes.**get_current_categories**()

Endpoint to retrieve categories with at least one answer from the current user.

Returns:

Response: A JSON object containing the IDs and names of categories with at least one answer from the current user.

routes.settings.routes.**get_empty_categories**()

Endpoint to retrieve categories with no answers from the current user.

Returns:

Response: A JSON object containing the IDs and names of categories with no answers from the current user.

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

r

INDEX

U

V