[2]p()*1/2-^ [1]p1-^ TJ
3 10000pt

Area_server

Napte

Nov 05, 2023

Contents:

Chapter 1

ResetPassword Model

1.1 Model

class backendNew.models.ResetPassword.**ResetPassword**

Model for storing reset password tokens.

Parameters

- **id** (*int*) – The primary key for the ResetPassword instance.
- **user_id** (*int*) – The user ID associated with the token.
- **token** (*str*) – The reset password token (max length: 255 characters).
- **created_at** (*datetime*) – The timestamp when the token was created (auto-generated).

1.1.1 Meta

db_table

The database table name for the ResetPassword model.

1.1.2 Methods

to_dict()

Convert the ResetPassword object to a dictionary.

Returns
A dictionary containing the object's attributes.

Return type
dict

1.2 Attributes

- `id`: The primary key for the ResetPassword instance.
- `user_id`: The user ID associated with the token.
- `token`: The reset password token (max length: 255 characters).
- `created_at`: The timestamp when the token was created (auto-generated).

1.3 Methods

- `to_dict()`: Convert the ResetPassword object to a dictionary.

  This method returns a dictionary with the following keys: - `id`: The primary key. - `user_id`: The user ID. - `token`: The reset password token. - `created_at`: The creation timestamp in the format 'YYYY-MM-DD HH:MM:SS'.

Chapter 2

RuleModel Model

2.1 Model

class backendNew.models.RuleModel.**RuleModel**

Model for storing rule information.

Parameters

- **id** (*int*) – The primary key for the RuleModel instance.
- **title** (*str*) – The title of the rule (max length: 255 characters, nullable).
- **description** (*str*) – The description of the rule (max length: 255 characters, nullable).
- **user_id** (*int*) – The user ID associated with the rule.
- **trigger_id** (*int*) – The ID of the trigger associated with the rule.
- **triggerToken** (*str*) – The trigger token (max length: 255 characters, nullable).
- **action_id** (*int*) – The ID of the action associated with the rule.
- **actionToken** (*str*) – The action token (max length: 255 characters, nullable).
- **created_at** (*datetime*) – The timestamp when the rule was created (auto-generated).

2.1.1 Meta

db_table

The database table name for the RuleModel model.

2.1.2 Methods

`to_dict()`

Convert the RuleModel object to a dictionary.

Returns
A dictionary containing the object's attributes.

Return type
dict

2.2 Attributes

- `id`: The primary key for the RuleModel instance.
- `title`: The title of the rule (max length: 255 characters, nullable).
- `description`: The description of the rule (max length: 255 characters, nullable).
- `user_id`: The user ID associated with the rule.
- `trigger_id`: The ID of the trigger associated with the rule.
- `triggerToken`: The trigger token (max length: 255 characters, nullable).
- `action_id`: The ID of the action associated with the rule.
- `actionToken`: The action token (max length: 255 characters, nullable).
- `created_at`: The timestamp when the rule was created (auto-generated).

2.3 Methods

- `to_dict()`: Convert the RuleModel object to a dictionary.

  This method returns a dictionary with the following keys: - `id`: The primary key. - `title`: The title of the rule (or *None* if not specified). - `description`: The description of the rule (or *None* if not specified). - `user_id`: The user ID. - `trigger_id`: The trigger ID. - `triggerToken`: The trigger token (or *None* if not specified). - `action_id`: The action ID. - `actionToken`: The action token (or *None* if not specified). - `created_at`: The creation timestamp in the format 'YYYY-MM-DD HH:MM:SS'.

Chapter 3

ServiceModel Model

3.1 Model

class backendNew.models.ServiceModel.**ServiceModel**

Model for storing service information.

Parameters

- **id** (*int*) – The primary key for the ServiceModel instance.
- **name** (*str*) – The name of the service (max length: 80 characters, not nullable).
- **action** (*str*) – The action related to the service (max length: 80 characters, not nullable).
- **type** (*str*) – The type of the service (max length: 80 characters, not nullable).

3.1.1 Meta

db_table

The database table name for the ServiceModel model.

3.1.2 Methods

to_dict()

Convert the ServiceModel object to a dictionary.

Returns
A dictionary containing the object's attributes.

Return type
dict

3.2 Attributes

- `id`: The primary key for the ServiceModel instance.
- `name`: The name of the service (max length: 80 characters, not nullable).
- `action`: The action related to the service (max length: 80 characters, not nullable).
- `type`: The type of the service (max length: 80 characters, not nullable).

3.3 Methods

- `to_dict()`: Convert the ServiceModel object to a dictionary.

  This method returns a dictionary with the following keys: - `id`: The primary key. - `name`: The name of the service. - `action`: The action related to the service. - `type`: The type of the service.

Chapter 4

TokensModel Model

4.1 Model

class backendNew.models.TokensModel.**TokensModel**

Model for storing user tokens for various services.

Parameters

- **user_id** (*int*) – The primary key for the TokensModel instance.
- **googleToken** (*str*) – The Google token (max length: 255 characters, nullable).
- **twitterToken** (*str*) – The Twitter token (max length: 255 characters, nullable).
- **githubToken** (*str*) – The GitHub token (max length: 255 characters, nullable).
- **discordToken** (*str*) – The Discord token (max length: 255 characters, nullable).
- **spotifyToken** (*str*) – The Spotify token (max length: 500 characters, nullable).
- **linkedinToken** (*str*) – The LinkedIn token (max length: 500 characters, nullable).
- **trelloToken** (*str*) – The Trello token (max length: 500 characters, nullable).

4.1.1 Meta

db_table

The database table name for the TokensModel model.

4.1.2 Methods

to_dict()

Convert the TokensModel object to a dictionary.

Returns

A dictionary containing the object's attributes.

Return type

dict

4.2 Attributes

- `user_id`: The primary key for the TokensModel instance.
- `googleToken`: The Google token (max length: 255 characters, nullable).
- `twitterToken`: The Twitter token (max length: 255 characters, nullable).
- `githubToken`: The GitHub token (max length: 255 characters, nullable).
- `discordToken`: The Discord token (max length: 255 characters, nullable).
- `spotifyToken`: The Spotify token (max length: 500 characters, nullable).
- `linkedinToken`: The LinkedIn token (max length: 500 characters, nullable).
- `trelloToken`: The Trello token (max length: 500 characters, nullable).

4.3 Methods

- `to_dict()`: Convert the TokensModel object to a dictionary.

  This method returns a dictionary with the following keys: - `user_id`: The primary key. - `googleToken`: The Google token. - `twitterToken`: The Twitter token. - `githubToken`: The GitHub token. - `discordToken`: The Discord token. - `spotifyToken`: The Spotify token. - `linkedinToken`: The LinkedIn token. - `trelloToken`: The Trello token.

Chapter 5

UserToken Model

5.1 Model

class backendNew.models.UserToken.**UserToken**

Model for storing user tokens associated with UserModel.

Parameters

- **id** (*int*) – The primary key for the UserToken instance.
- **token** (*str*) – The user token (max length: 80 characters, unique).
- **created_at** (*datetime*) – The timestamp when the token was created.
- **user_id** (*int*) – The foreign key to UserModel.

5.1.1 Meta

db_table

The database table name for the UserToken model.

5.1.2 Methods

to_dict()

Convert the UserToken object to a dictionary.

Returns
A dictionary containing the object's attributes.

Return type
dict

5.2 Attributes

- `id`: The primary key for the UserToken instance.
- `token`: The user token (max length: 80 characters, unique).
- `created_at`: The timestamp when the token was created.
- `user_id`: The foreign key to UserModel.

5.3 Methods

- `to_dict()`: Convert the UserToken object to a dictionary.

  This method returns a dictionary with the following keys: - `id`: The primary key. - `token`: The user token. - `created_at`: The timestamp when the token was created. - `user`: The associated user (UserModel instance).

Chapter 6

UserModel Model

6.1 Model

class backendNew.models.UserModel.**UserModel**

Model for storing user information.

Parameters

- **id** (*int*) – The primary key for the UserModel instance.
- **name** (*str*) – The user's name (max length: 80 characters, not null).
- **surname** (*str*) – The user's surname (max length: 80 characters, not null).
- **email** (*str*) – The user's email address (max length: 80 characters, unique).
- **password** (*str*) – The user's password (max length: 255 characters, not null).
- **created_at** (*datetime*) – The timestamp when the user was created.

6.1.1 Meta

db_table

The database table name for the UserModel model.

6.1.2 Methods

to_dict(*excludePassword: bool = False*)

Convert the UserModel object to a dictionary.

Parameters

excludePassword (*bool*) – If True, the 'password' field will be excluded from the dictionary.

Returns

A dictionary containing the object's attributes.

Return type

dict

6.2 Attributes

- `id`: The primary key for the UserModel instance.
- `name`: The user's name (max length: 80 characters, not null).
- `surname`: The user's surname (max length: 80 characters, not null).
- `email`: The user's email address (max length: 80 characters, unique).
- `password`: The user's password (max length: 255 characters, not null).
- `created_at`: The timestamp when the user was created.

6.3 Methods

- `to_dict(excludePassword: bool = False)`: Convert the UserModel object to a dictionary.

  This method returns a dictionary with the following keys: - `id`: The primary key. - `name`: The user's name. - `surname`: The user's surname. - `email`: The user's email address. - `password`: (optional) The user's password (excluded if excludePassword is True).

Chapter 7

About Functions

7.1 getAbout View

This view provides information about the application and its services.

7.1.1 View Details

View Name: getAbout

View Function: *getAbout(request) -> HttpResponse*

7.1.2 Parameters

- `request` (`HttpRequest`): The HTTP request object.

7.1.3 Return Value

- (`HttpResponse`): HTTP response containing application information.

7.1.4 View Usage

This view is designed to return information about the application and its services. When a GET request is made to this view, it will respond with a JSON document containing details about the application's services.

The response JSON format includes:

- A client section with host information.
- A server section with the current time and a list of services.

Chapter 8

Areas Functions

create View

Create a new rule based on the provided data.

8.1 View Function

create(request) -> HttpResponse

8.1.1 View Details

Parameters: - `request` (`HttpRequest`): The HTTP request object.

Return Value: - (`HttpResponse`): HTTP response containing the created rule data.

listByUser View

List rules created by the authenticated user.

8.2 View Function

listByUser(request) -> HttpResponse

8.2.1 View Details

Parameters: - `request` (`HttpRequest`): The HTTP request object.

Return Value: - (`HttpResponse`): HTTP response containing a list of rules.

getServicesActions View

Get actions provided by specific services.

8.3 View Function

getServicesActions(request) -> HttpResponse

8.3.1 View Details

Parameters: - `request` (`HttpRequest`): The HTTP request object.

Return Value: - (`HttpResponse`): HTTP response containing a list of service actions.

getServices View

Get unique service names based on the provided type.

8.4 View Function

getServices(request) -> HttpResponse

8.4.1 View Details

Parameters: - `request` (`HttpRequest`): The HTTP request object.

Return Value: - (`HttpResponse`): HTTP response containing unique service names.

getById View

Get, delete, or update a rule based on the provided ID.

8.5 View Function

getById(request, _id) -> HttpResponse

8.5.1 View Details

Parameters: - `request` (`HttpRequest`): The HTTP request object. - `_id` (int): The ID of the rule to retrieve, delete, or update.

Return Value: - (`HttpResponse`): HTTP response containing the rule data or operation status.

Chapter 9

CheckForReactions Functions

checkForReactions Function

Check for reactions and generate corresponding URLs based on the provided parameters.

9.1 Function Signature

checkForReactions(app, event, content, triggerToken) -> str or None

9.1.1 Function Details

Parameters: - `app` (str): The application or service name. - `event` (str): The event or action. - `content` (str): The content or data associated with the event. - `triggerToken` (str): The trigger token for authorization.

Return Value: - (str or None): The URL generated based on the reaction, or None if no matching reaction is found.

Chapter 10

Discord Functions

Login Function

Redirects the user to the Discord OAuth login page.

10.1 Function Signature

login(request) -> django.shortcuts.redirect

10.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the Discord OAuth login page.

Login Profile Function

Redirects the user to the Discord OAuth login page for profile data.

10.2 Function Signature

login_profile(request) -> django.shortcuts.redirect

10.2.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the Discord OAuth login page for profile data.

Callback Function

Handles the callback after a user logs in via Discord OAuth.

10.3 Function Signature

callback(request) -> django.shortcuts.redirect

10.3.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the client with an access token.

Callback Profile Function

Handles the callback for profile data after a user logs in via Discord OAuth.

10.4 Function Signature

callback_profile(request) -> django.shortcuts.redirect

10.4.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the client with an access token.

SetToken Function

Sets or updates a Discord token for the user.

10.5 Function Signature

setToken(request) -> django.http.HttpResponse

10.5.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the token information or an error response.

GetGuilds Function

Gets the guilds (servers) the user is a member of.

10.6 Function Signature

getGuilds(request) -> django.http.HttpResponse

10.6.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the list of guilds or an error response.

UpdateStatus Function

Handles the user's status update event.

10.7 Function Signature

updateStatus(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.7.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

UserTyping Function

Handles the user typing event in a channel.

10.8 Function Signature

userTyping(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.8.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

UserJoin Function

Handles the user joining a server event.

10.9 Function Signature

userJoin(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.9.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

UserRemove Function

Handles the user being removed from a server event.

10.10 Function Signature

userRemove(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.10.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

UserBan Function

Handles the user being banned from a server event.

10.11 Function Signature

userBan(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.11.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

UserUnban Function

Handles the user being unbanned from a server event.

10.12 Function Signature

userUnban(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.12.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

MessageChannel Function

Handles the event of a message being sent to a channel.

10.13 Function Signature

messageChannel(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.13.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

ChannelCreate Function

Handles the event of a new channel being created.

10.14 Function Signature

channelCreate(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.14.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

ChannelUpdate Function

Handles the event of an existing channel being updated.

10.15 Function Signature

channelUpdate(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.15.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

ChannelDelete Function

Handles the event of a channel being deleted.

10.16 Function Signature

channelDelete(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.16.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

PinsUpdate Function

Handles the event of a message being pinned in a channel.

10.17 Function Signature

pinsUpdate(request) -> django.shortcuts.redirect or django.http.HttpResponse

10.17.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect` or `django.http.HttpResponse`): A redirection to a reaction path or a success response.

Chapter 11

Github Functions

Login Function

Redirects the user to the GitHub OAuth login page.

11.1 Function Signature

login(request) -> django.shortcuts.redirect

11.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the GitHub OAuth login page.

Login Profile Function

Redirects the user to the GitHub OAuth login page for profile data.

11.2 Function Signature

login_profile(request) -> django.shortcuts.redirect

11.2.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the GitHub OAuth login page for profile data.

GetUser Function

Retrieves user data from GitHub.

11.3 Function Signature

getUser(request) -> django.http.HttpResponse

11.3.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with user data or an error response.

RedirectToRoot Function

Handles the GitHub OAuth callback and redirects to the root page.

11.4 Function Signature

redirectToRoot(request) -> django.shortcuts.redirect

11.4.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response with the user's access token.

Callback Profile Function

Handles the GitHub OAuth callback for profile data and redirects to the client with an access token.

11.5 Function Signature

callback_profile(request) -> django.shortcuts.redirect

11.5.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response with the user's access token.

SetToken Function

Sets or updates a GitHub token for the user.

11.6 Function Signature

setToken(request) -> django.http.HttpResponse

11.6.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the token information or an error response.

CreateRepo Function

Creates a new GitHub repository for the user.

11.7 Function Signature

createRepo(request) -> django.http.HttpResponse

11.7.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the repository creation status or an error response.

DeleteRepo Function

Deletes a GitHub repository for the user.

11.8 Function Signature

deleteRepo(request) -> django.http.HttpResponse

11.8.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the repository deletion status or an error response.

GetListRepo Function

Retrieves a list of GitHub repositories for the specified user.

11.9 Function Signature

getListRepo(request) -> django.http.HttpResponse

11.9.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the list of repositories or an error response.

Chapter 12

Google Functions

Login Function

Redirects the user to the Google OAuth login page based on the device type.

12.1 Function Signature

login(request) -> django.shortcuts.redirect

12.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the Google OAuth login page.

Login Profile Function

Redirects the user to the Google OAuth login page for profile data.

12.2 Function Signature

login_profile(request) -> django.shortcuts.redirect

12.2.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the Google OAuth login page for profile data.

Callback Function

Handles the Google OAuth callback and redirects to the client with an access token.

12.3 Function Signature

callback(request) -> django.shortcuts.redirect

12.3.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response with the user's access token.

Callback Profile Function

Handles the Google OAuth callback for profile data and redirects to the client with an access token.

12.4 Function Signature

callback_profile(request) -> django.shortcuts.redirect

12.4.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response with the user's access token.

RedirectToRoot Function

Handles the Google OAuth callback and redirects to the client with an access token.

12.5 Function Signature

redirectToRoot(request) -> django.shortcuts.redirect

12.5.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response with the user's access token.

SetToken Function

Sets or updates a Google token for the user.

12.6 Function Signature

setToken(request) -> django.http.HttpResponse

12.6.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the token information or an error response.

GetAllCalendar Function

Retrieves the user's Google Calendar data.

12.7 Function Signature

getAllCalendar(request) -> django.http.HttpResponse

12.7.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the Google Calendar data or an error response.

CreateFile Function

Creates a Google Drive file with the specified title.

12.8 Function Signature

create_file(title, googleToken) -> str

12.8.1 Function Details

Parameters: - `title`: The title of the file. - `googleToken`: The user's Google token.

Return Value: - (`str`): The ID of the created Google Drive file.

AddNewRow Function

Adds a new row to a Google Sheets document.

12.9 Function Signature

add_new_row(request) -> django.http.HttpResponse

12.9.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with a success message or an error response.

SendEmail Function

Sends an email to the specified recipient.

12.10 Function Signature

sendEmail(request) -> django.http.HttpResponse

12.10.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with a success message or an error response.

Chapter 13

Linkedin Functions

Login Profile Function

Redirects the user to the LinkedIn OAuth login page.

13.1 Function Signature

login_profile(request) -> django.shortcuts.redirect

13.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the LinkedIn OAuth login page.

Callback Function

Handles the LinkedIn OAuth callback and retrieves the access token.

13.2 Function Signature

callback(request) -> django.shortcuts.redirect or django.http.JsonResponse

13.2.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response with the user's access token. - (`django.http.JsonResponse`): JSON response in case of an error.

SetToken Function

Sets or updates a LinkedIn token for the user.

13.3 Function Signature

setToken(request) -> django.http.HttpResponse

13.3.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the token information or an error response.

Post a LinkedIn Post Function

Posts a message to LinkedIn on behalf of the user.

13.4 Function Signature

postALinkedinPost(request) -> django.http.HttpResponse

13.4.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with a success message or an error response.

Chapter 14

Spotify Functions

Login Function

Redirects the user to the Spotify OAuth login page.

14.1 Function Signature

login(request) -> django.shortcuts.redirect

14.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the Spotify OAuth login page.

Login Profile Function

Redirects the user to the Spotify OAuth login page for the profile.

14.2 Function Signature

login_profile(request) -> django.shortcuts.redirect

14.2.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response to the Spotify OAuth login page for the profile.

Callback Function

Handles the Spotify OAuth callback and retrieves the access token.

14.3 Function Signature

callback(request) -> django.shortcuts.redirect or django.http.JsonResponse

14.3.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response with the user's access token. - (`django.http.JsonResponse`): JSON response in case of an error.

Callback Profile Function

Handles the Spotify OAuth callback for the profile and retrieves the access token.

14.4 Function Signature

callback_profile(request) -> django.shortcuts.redirect or django.http.JsonResponse

14.4.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirect response with the user's access token. - (`django.http.JsonResponse`): JSON response in case of an error.

SetToken Function

Sets or updates a Spotify token for the user.

14.5 Function Signature

setToken(request) -> django.http.HttpResponse

14.5.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the token information or an error response.

Follow Playlist Function

Follows a Spotify playlist on behalf of the user.

14.6 Function Signature

follow_playlist(request) -> django.http.HttpResponse

14.6.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): HTTP response with success or error status code.

Unfollow Playlist Function

Unfollows a Spotify playlist on behalf of the user.

14.7 Function Signature

unfollow_playlist(request) -> django.http.HttpResponse

14.7.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): HTTP response with success status code.

Follow Artist Function

Follows a Spotify artist on behalf of the user.

14.8 Function Signature

follow_artist(request) -> django.http.HttpResponse

14.8.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): HTTP response with success status code.

Unfollow Artist Function

Unfollows a Spotify artist on behalf of the user.

14.9 Function Signature

unfollow_artist(request) -> django.http.HttpResponse

14.9.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): HTTP response with success status code.

Follow User Function

Follows a Spotify user on behalf of the user.

14.10 Function Signature

follow_user(request) -> django.http.HttpResponse

14.10.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): HTTP response with success status code.

Unfollow User Function

Unfollows a Spotify user on behalf of the user.

14.11 Function Signature

unfollow_user(request) -> django.http.HttpResponse

14.11.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): HTTP response with success status code.

Save Track Function

Saves a Spotify track on behalf of the user.

14.12 Function Signature

save_track(request) -> django.http.HttpResponse

14.12.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): HTTP response with success status code.

Unsave Track Function

Unsaves a Spotify track on behalf of the user.

14.13 Function Signature

unsave_track(request) -> django.http.HttpResponse

14.13.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): HTTP response with success status code.

Chapter 15

Status Function

Retrieve the status of user-connected services.

15.1 Function Signature

getStatus(request) -> django.http.JsonResponse

15.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.JsonResponse`): JSON response with the status of user-connected services.

Chapter 16

Trello Functions

Login Profile Function

Redirects the user to the Trello authorization page for profile access.

16.1 Function Signature

login_profile(request) -> django.shortcuts.redirect

16.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirects the user to the Trello authorization page.

16.2 Function Description

This function initiates the Trello OAuth authorization flow to allow the user to grant access to their Trello profile. It constructs the Trello authorization URL with the necessary parameters and redirects the user to that URL.

SetToken Function

Set and store the Trello access token for the user.

16.3 Function Signature

setToken(request) -> django.http.HttpResponse

16.3.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the stored Trello access token.

Chapter 17

Twitter Functions

Login Function

Redirects the user to the Twitter authorization page for standard access.

17.1 Function Signature

login(request) -> django.shortcuts.redirect

17.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirects the user to the Twitter authorization page.

17.2 Function Description

This function initiates the Twitter OAuth authorization flow to allow the user to grant access to their Twitter account for standard access. It constructs the Twitter authorization URL with the necessary parameters and redirects the user to that URL.

Login Profile Function

Redirects the user to the Twitter authorization page for profile access.

17.3 Function Signature

login_profile(request) -> django.shortcuts.redirect

17.3.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirects the user to the Twitter authorization page.

17.4 Function Description

This function initiates the Twitter OAuth authorization flow to allow the user to grant access to their Twitter account for profile access. It constructs the Twitter authorization URL with the necessary parameters and redirects the user to that URL.

Callback Function

Handle the callback after Twitter authorization.

17.5 Function Signature

callback(request) -> django.shortcuts.redirect or django.http.JsonResponse

17.5.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirects the user to the specified endpoint with the access token. - (`django.http.JsonResponse`): Returns an error response.

17.6 Function Description

This function handles the callback from Twitter after user authorization and stores the Twitter access token in the database. It first checks for errors and returns an error response if an error occurred.

If the authorization is successful, it extracts the authorization code and email, and uses them to obtain an access token from Twitter. If the access token is obtained successfully, it is stored in the database. The function then redirects the user to a specified endpoint with the access token, or returns an error response if there are issues.

Callback Profile Function

Handle the callback after Twitter authorization for profile access.

17.7 Function Signature

callback_profile(request) -> django.shortcuts.redirect or django.http.JsonResponse

17.7.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.shortcuts.redirect`): Redirects the user to the specified endpoint with the access token. - (`django.http.JsonResponse`): Returns an error response.

17.8 Function Description

This function handles the callback from Twitter after user authorization for profile access. It is similar to the *callback* function but is designed for profile access. It checks for errors and returns an error response if an error occurred.

If the authorization is successful, it extracts the authorization code and obtains an access token from Twitter. If the access token is obtained successfully, it redirects the user to a specified endpoint with the access token, or returns an error response if there are issues.

Auth Function

Authenticate user and obtain Twitter tokens.

17.9 Function Signature

auth(request) -> django.http.HttpResponse

17.9.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the Twitter tokens and user profile information.

Chapter 18

User Authentication and Password Reset API

Signup Function

User registration endpoint.

18.1 Function Signature

signup(request) -> django.http.JsonResponse

18.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.JsonResponse`): JSON response with the user's token or an error response.

18.2 Function Description

This function allows users to sign up by providing their name, surname, email, and password. If the email is unique, a user account is created. It expects a POST request with JSON data containing user details.

If the provided email is unique, a new user is created with the given details, and a unique token is generated for the user. The function returns a JSON response with the generated token. If the email already exists, it returns an error response.

Signin Function

User login endpoint.

18.3 Function Signature

signin(request) -> django.http.JsonResponse

18.3.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.JsonResponse`): JSON response with the user's token or an error response.

18.4 Function Description

This function allows users to sign in by providing their email and password. If the credentials are correct, a new token is generated for the user. It expects a POST request with JSON data containing the user's email and password.

If the provided credentials are correct, a new token is generated and associated with the user. The function returns a JSON response with the generated token. If the credentials are invalid, it returns an error response.

Signup Auth Function

User authentication for sign-up with an external provider.

18.5 Function Signature

signup_auth(request) -> django.http.JsonResponse

18.5.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.JsonResponse`): JSON response with the user's token or an error response.

18.6 Function Description

This function allows users to sign up using an external provider (e.g., Twitter) by providing their name, email, token, and the provider name. It expects a POST request with JSON data containing user details.

Depending on the external provider (e.g., Twitter), it validates the provided token and creates a new user if the authentication is successful. The function returns a JSON response with the generated token. If the authentication fails, it returns an error response.

Signin Auth Function

User authentication for sign-in with an external provider.

18.7 Function Signature

signin_auth(request) -> django.http.JsonResponse

18.7.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.JsonResponse`): JSON response with the user's token or an error response.

18.8 Function Description

This function allows users to sign in using an external provider (e.g., Twitter) by providing their email and token. It expects a POST request with JSON data containing user details.

Depending on the external provider (e.g., Twitter), it validates the provided token and generates a new token for the user. The function returns a JSON response with the generated token. If the authentication fails, it returns an error response.

User Exists Function

Check if a user exists based on their email.

18.9 Function Signature

user_exists(request) -> django.http.HttpResponse

18.9.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the user's details or an error response.

18.10 Function Description

This function checks if a user exists in the database based on their email. It expects a POST request with JSON data containing the user's email.

If the user exists, it returns a JSON response with the user's details. If the user does not exist, it returns an error response.

SendMail Function

Send a password reset email.

18.11 Function Signature

sendMail(request) -> django.http.JsonResponse

18.11.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.JsonResponse`): JSON response indicating success or an error response.

18.12 Function Description

This function sends a password reset email to the user's email address. It expects a POST request with JSON data containing the user's email.

The function generates a unique token and sends a password reset email to the user's email address. It returns a JSON response indicating success. If there's an error, it returns an error response.

ResetPassword Function

Reset the user's password.

18.13 Function Signature

resetPassword(request, token) -> django.http.JsonResponse

18.13.1 Function Details

Parameters: - `request`: The HTTP request object. - `token`: Reset password token.

Return Value: - (`django.http.JsonResponse`): JSON response indicating success or an error response.

18.14 Function Description

This function resets the user's password based on a provided token. It expects a POST request with JSON data containing the new password and the token.

If the provided token is valid and not expired, it changes the user's password and deletes the token. The function returns a JSON response indicating success. If the token is invalid or expired, it returns an error response.

IsTokenValid Function

Check if the user's authentication token is valid.

18.15 Function Signature

isTokenValid(request) -> django.http.HttpResponse

18.15.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response with the user's details or an error response.

18.16 Function Description

This function checks if the user's authentication token is valid for the given provider (e.g., Twitter or Google). It expects a POST request with JSON data containing the provider name (e.g., Twitter or Google).

Depending on the provider, it validates the user's authentication token and returns a JSON response with the user's details. If the token is invalid or the provider is not recognized, it returns an error response.

Me Function

Get the user's profile details

Chapter 19

GitHub Webhook Data Handling API

Receive GitHub Function

Receive GitHub webhook data.

19.1 Function Signature

receiveGitHub(request) -> django.http.HttpResponse

19.1.1 Function Details

Parameters: - `request`: The HTTP request object.

Return Value: - (`django.http.HttpResponse`): JSON response or HTTP status code indicating the result of processing the GitHub webhook data.

19.2 Function Description

This function handles incoming GitHub webhook data, parses it using the GitHubParser, and checks for reactions using checkForReactions.

It expects a POST request with JSON data containing GitHub webhook information.

- If the provided data is parsed successfully and contains valid action and message information, it checks for reactions using the "checkForReactions" function.
- If "checkForReactions" returns a path (i.e., a URL or message), the function returns a JSON response with the path and an HTTP status code of 200.
- If the provided action or message is "default," it returns an HTTP response with a "Not Implemented" status code of 501.
- If the request method is not POST, it returns an HTTP response with an "Invalid Method" status code of 405.
- For any other errors or exceptions, it returns an HTTP response with an internal server error status code of 500.

Chapter 20

Githubparser Class

class backendNew.utils.githubparser.**GithubParser**

Class for parsing Github webhooks responses.

static parseWebhook(*resp*)

This method parses a Github webhook response and returns a dictionary with action and message information.

Parameters
resp (*dict*) – The Github webhook response to parse.

Returns
A dictionary containing the action and message.

Return type
dict

The possible actions and their corresponding messages are as follows:

- When a user pushes to a repository: "When a user push"
- When a user creates a repository: "When a user create a repository"
- When a user opens a pull request: "When a user open a pull request"
- When a user closes a pull request: "When a user close a pull request"
- When a user creates a branch: "When a user create a branch"
- When a user merges a branch: "When a user merge a branch"

20.1 Githubparser Class

The *Githubparser* class is used for parsing Github webhooks responses.

20.2 GithubParser.parseWebhook Method

static GithubParser.**parseWebhook**(*resp*)

This method parses a Github webhook response and returns a dictionary with action and message information.

Parameters
resp (*dict*) – The Github webhook response to parse.

Returns
A dictionary containing the action and message.

Return type
dict

The possible actions and their corresponding messages are as follows:

- When a user pushes to a repository: "When a user push"
- When a user creates a repository: "When a user create a repository"
- When a user opens a pull request: "When a user open a pull request"
- When a user closes a pull request: "When a user close a pull request"
- When a user creates a branch: "When a user create a branch"
- When a user merges a branch: "When a user merge a branch"

This method parses a Github webhook response and returns a dictionary with action and message information.

Parameters:

- *resp* (dict): The Github webhook response to parse.

Returns:

- A dictionary containing the action and message.

Possible actions and their corresponding messages:

- When a user pushes to a repository: "When a user push"
- When a user creates a repository: "When a user create a repository"
- When a user opens a pull request: "When a user open a pull request"
- When a user closes a pull request: "When a user close a pull request"
- When a user creates a branch: "When a user create a branch"
- When a user merges a branch: "When a user merge a branch"

Chapter 21

Token Functions

21.1 Module Overview

This module provides functions for generating and managing tokens. It includes the following functions:

- `generateToken()`: Generates a random token with a specific format.
- `check_token()`: Checks if a token is still valid.
- `analyseToken()`: Analyzes a token from a request.

21.2 Functions

GenerateToken Function

Generates a random token with a specific format.

21.3 Function Signature

generateToken(size = 32, indexes = [7, 11, 15, 19]) -> str

21.3.1 Function Details

Parameters: - `size`: The size of the token. - `indexes`: The indexes where you want the '-' to be.

Return Value: - (`str`): The newly generated token.

Check_token Function

Checks if a token is still valid.

21.4 Function Signature

check_token(token, minutes=30) -> bool

21.4.1 Function Details

Parameters: - `token`: The token to check. - `minutes`: The time you want the tokens to be valid.

Return Value: - (`bool`): True if the token is valid, False otherwise.

AnalyseToken Function

Checks if a token is still valid.

21.5 Function Signature

analyseToken(request) -> str or None

21.5.1 Function Details

Parameters: - `request`: The Http request.

Return Value: - (`str`): The token if it is valid, None otherwise.

Chapter 22

Indices and tables

- genindex
- modindex
- search

Python Module Index

b

Index