Wirespace

Generated by Doxygen 1.8.11

Contents

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 share Namespace Reference

Namespaces

- admin
- apps
- middleware
- models
- urls
- views

5.2 share.admin Namespace Reference

Classes

- class KeyAdmin

  Class that handles how the Key section would look on the admin page.
- class TokenAdmin

  Class that handles how the Token section would look on the admin page.

5.3 share.apps Namespace Reference

Classes

- class ShareConfig

  Class to manage app configuration.

5.4 share.middleware Namespace Reference

Classes

- class AuthenticateTokenMiddleware

  Middleware class to handle token authentication.
- class ExpireKeyMiddleware

  Middleware class to handle expiration of keys.

Variables

- list CHECK_LIST

  List of all paths for which the request has to be validated.
- list POST_IGNORE_LIST

  List of paths for which the POST requests have to be ignored during validation.
- re_host = re.compile('/host[/]?.∗')

  Regex for host/.
- re_editor = re.compile('/editor[/]?.∗')

  Regex for editor/.
- re_key = re.compile('/[0-9a-f]{16}[/]?$')

  Regex to match authentication keys.

5.4.1 Variable Documentation

5.4.1.1 list share.middleware.CHECK_LIST

Initial value:

```
= [
    '/share/',
    '/share/open/',
    '/share/upload/',
    '/share/uploadFolder/',
    '/share/download/',
    '/share/delete/',
    '/share/create_folder/',
    '/share/search/',
    '/share/move/',
]
```

List of all paths for which the request has to be validated.

5.4.1.2 list share.middleware.POST_IGNORE_LIST

Initial value:

```
= [
    '/share/',
]
```

List of paths for which the POST requests have to be ignored during validation.

5.4.1.3 share.middleware.re_editor = re.compile('/editor[/]?.∗')

Regex for editor/.

5.4.1.4 share.middleware.re_host = re.compile('/host[/]?.∗')

Regex for host/.

5.4.1.5 share.middleware.re_key = re.compile('/[0-9a-f]{16}[/]?$')

Regex to match authentication keys.

5.5 share.models Namespace Reference

Classes

- class Key

  Class to store all the information about the shared space.
- class Token

  Class representing tokens for one-time-authentication using key.

Functions

- def gen_key (length=8)

  Function to generate a random hex-string of given length.
- def gen_token (length=16)

  Function to generate a unique token of specifed length.

5.5.1 Function Documentation

5.5.1.1 def share.models.gen_key (*length* = 8)

Function to generate a random hex-string of given length.

This function keeps on generating a new string until it finds one that does not exist in the database.

Parameters

length	The length of the hexstring to be generated with default as 8

Returns

A random, unique, hex-string of specified length

5.5.1.2 def share.models.gen_token (*length* = `16`)

Function to generate a unique token of specifed length.

Parameters

length	The length of the token to be generated with default as 16

Returns

A string representing the unique, hexadecimal token generated

5.6 share.urls Namespace Reference

Variables

- list urlpatterns

  List that contains matching regex for urls and corresponding view functions to be invoked.

5.6.1 Variable Documentation

5.6.1.1 list share.urls.urlpatterns

Initial value:

```
= [
    url(r'^$',views.home,name='home'),
        url(r'^open/$',views.open_item,name='open'),
        url(r'^download/$',views.download_item,name='download'),
        url(r'^search/$',views.search,name='search'),
        url(r'^upload/$',views.upload,name='upload'),
        url(r'^create_folder/$',views.create_folder,name='create_folder'),
        url(r'^delete/$',views.delete,name='delete'),
        url(r'^move/$',views.move,name='move'),
]
```

List that contains matching regex for urls and corresponding view functions to be invoked.

5.7 share.views Namespace Reference

Functions

- def authenticate (request, k)

  View to handle one-time authentication for the current session.
- def editor_authenticate (request, k)

  View to handle authentication for the file editor.
- def editor (request)

  View for handling the save, download, upload and editing features of the editor.
- def home (request)

View for opening the homepage.

- def get_file (filepath, mode)

  Function that downloads/opens the required file as per the request.
- def get_dir (dirpath)

  Function to download the specified directory.
- def open_item (request)

  View to handle opening of the selected directory/file.
- def download_item (request)

  View to download all/selected content of the current directory.
- def upload (request)

  View to upload a file or folder from the client.
- def search (request)

  View to search the current directory recursively for a file or folder.
- def delete (request)

  View to handle deletion of a file or folder on the server via a request from the client.
- def create_folder (request)

  View to create a new folder with a specified name.
- def move (request)

  A generic function for moving a file/directory from one place to another (can also be used for renaming)

Variables

- string CACHE_DIR = '/tmp/wirespace/cache'

  Used for storing generated .zip files.

5.7.1 Function Documentation

5.7.1.1 def share.views.authenticate (*request*, *k*)

View to handle one-time authentication for the current session.

Checks if the correct key has been provided and then creates a token which expires after 60 minutes.

Parameters

request	Instance of HttpRequest containing required meta-data
k	The key passed by the client

Returns

Depending on the success of the authentication, an error message is displayed or the page is redirected to share/

5.7.1.2 def share.views.create_folder (*request*)

View to create a new folder with a specified name.

If the folder does not exist previously, a new folder is created with the required name (The functionality is similar to mkdir -p). Also, entering incorrect folder names will not do anything.

Parameters

request	POST request containing the name of the folder to be created and its address

Returns

A JsonResponse object denoting success or failure of the action

5.7.1.3 def share.views.delete (*request*)

View to handle deletion of a file or folder on the server via a request from the client.

If the client has write priviledges, it checks if the target is a file or folder and then deletes it PERMANENTLY using respective python functions.

Parameters

request	POST request containing the address of the target to be deleted

Returns

A success or error message as a JsonResponse object

5.7.1.4 def share.views.download_item (*request*)

View to download all/selected content of the current directory.

If the directory contains just one item, the request is forwarded to the appropriate functions(*get_dir* or *get_file*). Otherwise, it creates a .zip file of the contents of the directory. Note: Hidden files are not sent via this method.

Parameters

request	POST request containing the target list.

Returns

A StreamingHttpResponse object containing a zip file or a single file depending on the request

5.7.1.5 def share.views.editor (*request*)

View for handling the save, download, upload and editing features of the editor.

Validates token and then performs different actions depending on the mode. These include serving file for editing, sending list of editable files, updating the edited file on the host's machine and destroying the associated token.

Parameters

request	Instance of HttpRequest containing required meta-data

Returns

A JsonResponse object containing a success or error message and relevant data

5.7.1.6 def share.views.editor_authenticate (*request, k*)

View to handle authentication for the file editor.

If the user has write permission and a correct key has been provided, the client is provided a unique token and a list of editable files.

Parameters

request	Instance of HttpRequest containing required meta-data
k	The key passed by the client

Returns

A JsonResponse object containing an error message, or a unique token and a list of files and directories for editing

5.7.1.7 def share.views.get_dir (*dirpath*)

Function to download the specified directory.

Compresses the directory into a zip file which is stored in *CACHE_DIR*, and returns a StreamingHttpResponse object

Parameters

dirpath	the full path of the directory to be accessed

Returns

A StreamingHttpResponse if no error occured, and a JsonResponse object otherwise

5.7.1.8 def share.views.get_file (*filepath, mode*)

Function that downloads/opens the required file as per the request.

Guesses the file type using mimetypes and then sets the 'Content-disposition' according to the request and sends back the response. If at any time, any error occurs, an error message is displayed.

Parameters

filepath	The full path of the file to be accessed
mode	Specifies whether the file has to be downloaded or opened in the browser

Returns

A StreamingHttpResponse object if no error occured, or a JsonResponse object otherwise

5.7.1.9 def share.views.home (*request*)

View for opening the homepage.

(/share/)

Parameters

request	Instance of HttpRequest containing required meta-data

5.7.1.10 def share.views.move (*request*)

A generic function for moving a file/directory from one place to another (can also be used for renaming)

If the client has write permission and the destination does not already exist, the files/folders are moved to the appropriate place. Otherwise, an error message is returned.

Parameters

request	POST request containing the source and target addresses

Returns

A message signifying whether the move was successful or not as a JsonResponse object.

5.7.1.11 def share.views.open_item (*request*)

View to handle opening of the selected directory/file.

The code checks if the target is a directory or a file. If it is a file, it redirects to the *get_file* function; otherwise it creates a dictionary of a list of contents in the directory and sends back the data as a JsonResponse object

Parameters

request	POST request containing the address of the target to be opened

Returns

A JsonResponse containing a list of files and folders present in the directory

5.7.1.12 def share.views.search (*request*)

View to search the current directory recursively for a file or folder.

Iterates through each file and directory recursively and checks if they contain the required substring. All hits (including hidden files) are returned as a JsonResponse object

Parameters

request	POST request containing the address of the directory to be searched and the search query

Returns

A JsonResponse object containing a list of all files and directories having the query in their name

5.7.1.13 def share.views.upload (*request*)

View to upload a file or folder from the client.

Checks if sufficient storage space is present and whether the client has editing rights. If the target path exists, the file is directly placed at the specified target. Otherwise, the target is first created and then the file is placed in it.

Parameters

request	POST request containing files with their addresses

Returns

A JsonResponse object containing a success or error message.

5.7.2 Variable Documentation

5.7.2.1 string share.views.CACHE_DIR = '/tmp/wirespace/cache'

Used for storing generated .zip files.

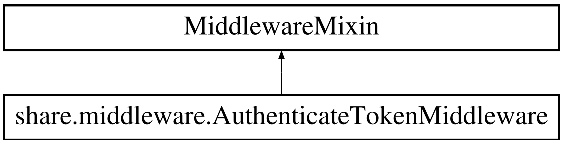
Chapter 6

Class Documentation

6.1 share.middleware.AuthenticateTokenMiddleware Class Reference

Middleware class to handle token authentication.

Inheritance diagram for share.middleware.AuthenticateTokenMiddleware:



Public Member Functions

- def process_request (self, request)

  Function to check whether the request contains a valid token or not.

6.1.1 Detailed Description

Middleware class to handle token authentication.

6.1.2 Member Function Documentation

6.1.2.1 def share.middleware.AuthenticateTokenMiddleware.process_request (*self, request*)

Function to check whether the request contains a valid token or not.

Only those requests which are specified in CHECK_LIST are validated.

Parameters

self	An instance of AuthenticateTokenMiddleware class
request	An HttpRequest object containing required metadata

Returns

None if no error occurs, a JsonResponse describing the error otherwise

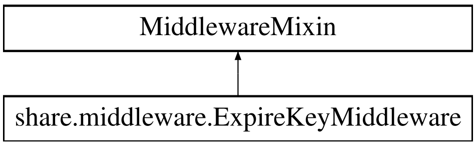
The documentation for this class was generated from the following file:

- middleware.py

6.2 share.middleware.ExpireKeyMiddleware Class Reference

Middleware class to handle expiration of keys.

Inheritance diagram for share.middleware.ExpireKeyMiddleware:



Public Member Functions

- def process_request (self, request)

  Function to check whether the key associated with the token in the request is still valid or not.

6.2.1 Detailed Description

Middleware class to handle expiration of keys.

6.2.2 Member Function Documentation

6.2.2.1 def share.middleware.ExpireKeyMiddleware.process_request (*self*, *request*)

Function to check whether the key associated with the token in the request is still valid or not.

If the key is found to be expired, it is deleted alongwith all the associated tokens and session objects.

Parameters

self	An instance of ExpireKeyMiddleware class
request	An HttpRequest object containing required metadata

Returns

None if no error occurs, a JsonResponse describing the error otherwise

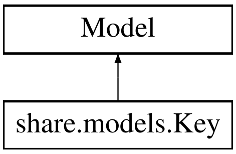
The documentation for this class was generated from the following file:

- middleware.py

6.3 share.models.Key Class Reference

Class to store all the information about the shared space.

Inheritance diagram for share.models.Key:



Public Member Functions

- def __str__ (self)

  Function to generate the string representation of an instance.
- def time_slot (self)

  Function to generate a string representation of the time interval within which the key is active.
- def space_available (self)

  Function to display the space avaiable for use.
- def link (self)

  Function to generate the shareable link.
- def clean (self)

  Function to validate the Key instance before it is saved It raises a ValidationError if some error in the to-be-created instance is detected.
- def save (self, args, kwargs)

  Function that overrides the default save function It fills optional, blank fields with default values before the instance is saved.

Static Public Attributes

- key = models.CharField(max_length=8,default=gen_key,editable=False, primary_key=True,help_text="This value is temporary. Use the final value after saving.")

  Stores the unique key associated with each instance that is provided to the client.
- permission = models.CharField(max_length=1,choices=(('r','Read-only'),('w','Read-and-Write')),default='r')

  The permission that the host gives to the client.
- shared_to = models.CharField(max_length=30,default="Anonymous")

  Name of the client to which the Key is shared.
- email = models.EmailField(max_length=254,help_text="E-mail of the person with whom the space is shared",null=True,blank=True)

  An optional field to store the email address of the client for sending notifications.
- space_allotted = models.BigIntegerField(help_text="Space to be shared, in BYTES",default=4096)

  The additional space provided by the host to the client apart from the already existing space in the specified path_↩shared.
- path_shared = models.TextField(default=os.path.expanduser("~/Desktop"),max_length=100)

  The shared path/directory associated with this instance.
- created_on = models.DateTimeField(auto_now_add=True)

  The timestamp when this instance was created.
- expires_on = models.DateTimeField(default=None)

  The date and time upto which the host wants to share his space.

6.3.1 Detailed Description

Class to store all the information about the shared space.

It saves these details to the database and helps in their retrieval upon clients' requests.

6.3.2 Member Function Documentation

6.3.2.1 def share.models.Key.__str__ (*self*)

Function to generate the string representation of an instance.

Parameters

self	An instance of Key class that needs to be represented

Returns

A string representation of *self*

6.3.2.2 def share.models.Key.clean (*self*)

Function to validate the Key instance before it is saved It raises a ValidationError if some error in the to-be-created instance is detected.

Parameters

self	An instance of Key class

6.3.2.3 def share.models.Key.link (*self*)

Function to generate the shareable link.

It concatenates the unique key associated with the instance with the host's IP.

Parameters

self	An instance of Key class

Returns

The shareable link to be shared with the client

6.3.2.4 def share.models.Key.save (*self, args, kwargs*)

Function that overrides the default save function It fills optional, blank fields with default values before the instance is saved.

It also adds the current space occupied by the *path_shared* directory to the *space_allotted* variable to account for the total space shared.

Parameters

self	The instance of Key class
∗*args*	Non-keyworded argument list
∗∗*kwargs*	Keyworded argument list

6.3.2.5 def share.models.Key.space_available (*self*)

Function to display the space avaiable for use.

Parameters

self	An instance of Key class

Returns

String representation of the space available for usage by the client in human-readable form

6.3.2.6 def share.models.Key.time_slot (*self*)

Function to generate a string representation of the time interval within which the key is active.

Parameters

self	An instance of Key class

Returns

A string representation of the time-interval for the specified instance

6.3.3 Member Data Documentation

6.3.3.1 share.models.Key.created_on = models.DateTimeField(auto_now_add=True) [static]

The timestamp when this instance was created.

Gets filled automatically during time of creation and cannot be altered.

6.3.3.2 share.models.Key.email = models.EmailField(max_length=254,help_text="E-mail of the person with whom the space is shared",null=True,blank=True) [static]

An optional field to store the email address of the client for sending notifications.

6.3.3.3 share.models.Key.expires_on = models.DateTimeField(default=None) [static]

The date and time upto which the host wants to share his space.

6.3.3.4 share.models.Key.key = models.CharField(max_length=8,default=gen_key,editable=False, primary_key=True,help_text="This value is temporary. Use the final value after saving.") [static]

Stores the unique key associated with each instance that is provided to the client.

6.3.3.5 share.models.Key.path_shared = models.TextField(default=os.path.expanduser("∼/Desktop"),max_length=100) [static]

The shared path/directory associated with this instance.

6.3.3.6 share.models.Key.permission = models.CharField(max_length=1,choices=(('r','Read-only'),('w','Read-and-Write')),default='r') [static]

The permission that the host gives to the client.

Can be one of 'read' or 'read-and-write'

6.3.3.7 share.models.Key.shared_to = models.CharField(max_length=30,default="Anonymous") [static]

Name of the client to which the Key is shared.

6.3.3.8 share.models.Key.space_allotted = models.BigIntegerField(help_text="Space to be shared, in BYTES",default=4096) [static]

The additional space provided by the host to the client apart from the already existing space in the specified *path_shared*.

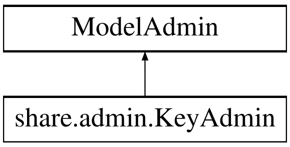
The documentation for this class was generated from the following file:

- models.py

6.4 share.admin.KeyAdmin Class Reference

Class that handles how the Key section would look on the admin page.

Inheritance diagram for share.admin.KeyAdmin:



Public Member Functions

- def get_readonly_fields (self, request, obj=None)

  Overridden function to specify which fields are read-only.

Static Public Attributes

- list fields = ['key','created_on','expires_on','path_shared','permission','space_allotted','shared_to','email']

  The fields to be included.
- tuple list_display = ('link','shared_to','path_shared','time_slot','space_available','permission')

  The fields to be displayed as columns in the tabular form.
- dictionary formfield_overrides

  Defines overriding widgets for specified fields.

6.4.1 Detailed Description

Class that handles how the Key section would look on the admin page.

6.4.2 Member Function Documentation

6.4.2.1 def share.admin.KeyAdmin.get_readonly_fields (*self*, *request*, *obj* = None)

Overridden function to specify which fields are read-only.

It converts the 'space_allotted' field to a read-only field once the Key object has been created.

Parameters

self	A KeyAdmin instance
request	An HttpRequest containing required metadata
obj	The Key object associated with the request

Returns

A list of read-only fields depending on the context

6.4.3 Member Data Documentation

6.4.3.1 list share.admin.KeyAdmin.fields = ['key','created_on','expires_on','path_shared','permission','space_allotted','shared_to','email'] [static]

The fields to be included.

6.4.3.2 dictionary share.admin.KeyAdmin.formfield_overrides [static]

Initial value:

```
= {
        models.TextField: {
            'widget': Textarea(
                attrs={'rows': 1,
                       'cols': 50,
                    })}}
```

Defines overriding widgets for specified fields.

6.4.3.3 tuple share.admin.KeyAdmin.list_display = ('link','shared_to','path_shared','time_slot','space_available','permission') [static]

The fields to be displayed as columns in the tabular form.

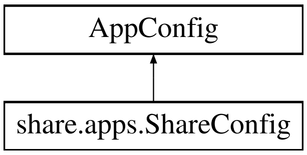
The documentation for this class was generated from the following file:

- admin.py

6.5 share.apps.ShareConfig Class Reference

Class to manage app configuration.

Inheritance diagram for share.apps.ShareConfig:



Static Public Attributes

- string name = 'share'

6.5.1 Detailed Description

Class to manage app configuration.

6.5.2 Member Data Documentation

6.5.2.1 string share.apps.ShareConfig.name = 'share' [static]

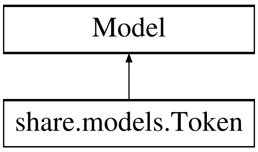
The documentation for this class was generated from the following file:

- apps.py

6.6 share.models.Token Class Reference

Class representing tokens for one-time-authentication using key.

Inheritance diagram for share.models.Token:



Public Member Functions

- def __str__ (self)

  Function to generate string representation of a Token instance.
- def save (self, args, kwargs)

  Function that overrides the default save function.

Static Public Attributes

- link = models.ForeignKey(Key,on_delete=models.CASCADE,default=None)

  Stores the Key instance associated with it.
- token = models.CharField(max_length=16,default=gen_token,editable=False,primary_key=True)

  The unique, auto-generated token assigned to a client on providing a valid key.
- IP = models.CharField(max_length=15,default=None)

  Stores the IP address of the client, if detected, for future reference.

6.6.1 Detailed Description

Class representing tokens for one-time-authentication using key.

Client data such as the associated key and IP address is stored in its instances.

6.6.2 Member Function Documentation

6.6.2.1 def share.models.Token.__str__ (*self*)

Function to generate string representation of a Token instance.

Parameters

self	A Token instance

Returns

The *token* attribute of the instance as a string

6.6.2.2 def share.models.Token.save (*self, args, kwargs*)

Function that overrides the default save function.

It first validates the fields, not saving the instance if any error is found. On successful validation, it saves the instance by calling the super save() function

Parameters

self	A Token instance
∗*args*	Non-keyworded arguments
∗∗*kwargs*	Keyworded arguments

6.6.3 Member Data Documentation

6.6.3.1 share.models.Token.IP = models.CharField(max_length=15,default=None) [static]

Stores the IP address of the client, if detected, for future reference.

6.6.3.2 share.models.Token.link = models.ForeignKey(Key,on_delete=models.CASCADE,default=None) [static]

Stores the Key instance associated with it.

6.6.3.3 share.models.Token.token = models.CharField(max_length=16,default=gen_token,editable=False,primary_key=True) [static]

The unique, auto-generated token assigned to a client on providing a valid key.

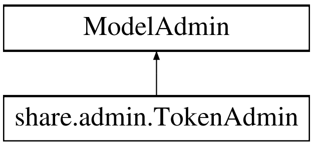
The documentation for this class was generated from the following file:

- models.py

6.7 share.admin.TokenAdmin Class Reference

Class that handles how the Token section would look on the admin page.

Inheritance diagram for share.admin.TokenAdmin:



Static Public Attributes

- tuple readonly_fields = ('link','token','IP')

  List of all read-only fields.
- list fields = ['link','token','IP']

  List of all fields to be included.
- tuple list_display = ('token','link')

  List of fields to be displayed as columns in the tabular form.

6.7.1 Detailed Description

Class that handles how the Token section would look on the admin page.

6.7.2 Member Data Documentation

6.7.2.1 list share.admin.TokenAdmin.fields = ['link','token','IP'] [static]

List of all fields to be included.

6.7.2.2 tuple share.admin.TokenAdmin.list_display = ('token','link') [static]

List of fields to be displayed as columns in the tabular form.

6.7.2.3 tuple share.admin.TokenAdmin.readonly_fields = ('link','token','IP') [static]

List of all read-only fields.

The documentation for this class was generated from the following file:

- admin.py

Chapter 7

File Documentation

7.1 __init__.py File Reference

Namespaces

- share

7.2 admin.py File Reference

This file defines how the admin page should look like.

Classes

- class share.admin.KeyAdmin

  Class that handles how the Key section would look on the admin page.
- class share.admin.TokenAdmin

  Class that handles how the Token section would look on the admin page.

Namespaces

- share.admin

7.2.1 Detailed Description

This file defines how the admin page should look like.

7.3 apps.py File Reference

This file configures the app.

Classes

- class share.apps.ShareConfig

  Class to manage app configuration.

Namespaces

- share.apps

7.3.1 Detailed Description

This file configures the app.

7.4 middleware.py File Reference

This file contains the description middleware used to authenticate token and expire keys.

Classes

- class share.middleware.AuthenticateTokenMiddleware

  Middleware class to handle token authentication.
- class share.middleware.ExpireKeyMiddleware

  Middleware class to handle expiration of keys.

Namespaces

- share.middleware

Variables

- list share.middleware.CHECK_LIST

  List of all paths for which the request has to be validated.
- list share.middleware.POST_IGNORE_LIST

  List of paths for which the POST requests have to be ignored during validation.
- share.middleware.re_host = re.compile('/host[/]?.∗')

  Regex for host/.
- share.middleware.re_editor = re.compile('/editor[/]?.∗')

  Regex for editor/.
- share.middleware.re_key = re.compile('/[0-9a-f]{16}[/]?$')

  Regex to match authentication keys.

7.4.1 Detailed Description

This file contains the description middleware used to authenticate token and expire keys.

7.5 models.py File Reference

This file contains the description of the Key and Token classes, which are used for authentication purposes in the project.

Classes

- class share.models.Key

  Class to store all the information about the shared space.
- class share.models.Token

  Class representing tokens for one-time-authentication using key.

Namespaces

- share.models

Functions

- def share.models.gen_key (length=8)

  Function to generate a random hex-string of given length.
- def share.models.gen_token (length=16)

  Function to generate a unique token of specifed length.

7.5.1 Detailed Description

This file contains the description of the Key and Token classes, which are used for authentication purposes in the project.

7.6 urls.py File Reference

This file contains the description and resolution of all urls required for the project.

Namespaces

- share.urls

Variables

- list share.urls.urlpatterns

  List that contains matching regex for urls and corresponding view functions to be invoked.

7.6.1 Detailed Description

This file contains the description and resolution of all urls required for the project.

7.7 views.py File Reference

This file contains the definitions of all the functions required by the project.

Namespaces

- share.views

Functions

- def share.views.authenticate (request, k)

  View to handle one-time authentication for the current session.
- def share.views.editor_authenticate (request, k)

  View to handle authentication for the file editor.
- def share.views.editor (request)

  View for handling the save, download, upload and editing features of the editor.
- def share.views.home (request)

  View for opening the homepage.
- def share.views.get_file (filepath, mode)

  Function that downloads/opens the required file as per the request.
- def share.views.get_dir (dirpath)

  Function to download the specified directory.
- def share.views.open_item (request)

  View to handle opening of the selected directory/file.
- def share.views.download_item (request)

  View to download all/selected content of the current directory.
- def share.views.upload (request)

  View to upload a file or folder from the client.
- def share.views.search (request)

  View to search the current directory recursively for a file or folder.
- def share.views.delete (request)

  View to handle deletion of a file or folder on the server via a request from the client.
- def share.views.create_folder (request)

  View to create a new folder with a specified name.
- def share.views.move (request)

  A generic function for moving a file/directory from one place to another (can also be used for renaming)

Variables

- string share.views.CACHE_DIR = '/tmp/wirespace/cache'

  Used for storing generated .zip files.

7.7.1 Detailed Description

This file contains the definitions of all the functions required by the project.