
jerrycan
Release 0.1.0

Hermann Krumrey

Sep 29, 2020

CONTENTS

1	jerrycan package	3
1.1	Subpackages	3
1.2	Submodules	13
1.3	jerrycan.Config module	13
1.4	jerrycan.base module	15
1.5	jerrycan.enums module	15
1.6	jerrycan.exceptions module	16
1.7	jerrycan.initialize module	16
1.8	jerrycan.wsgi module	16
1.9	Module contents	17
2	jerrycan	19
3	Indices and tables	21
	Python Module Index	23
	Index	25

Contents:

JERRYCAN PACKAGE

1.1 Subpackages

1.1.1 jerrycan.background package

Submodules

jerrycan.background.telegram module

`jerrycan.background.telegram.telegram_whoami()`

Specifies the background behaviour of the telegram bot By default, the bot listens to /whoami messages and answers with the telegram chat ID :return: None

Module contents

1.1.2 jerrycan.db package

Submodules

jerrycan.db.ApiKey module

class `jerrycan.db.ApiKey.ApiKey(*args, **kwargs)`

Bases: `jerrycan.db.ModelMixin.ModelMixin`, `sqlalchemy.ext.declarative.api.Model`

Model that describes the 'api_keys' SQL table An ApiKey is used for API access using HTTP basic auth

__init__(*args, **kwargs)

Initializes the Model :param args: The constructor arguments :param kwargs: The constructor keyword arguments

creation_time: `int`

The time at which this API key was created as a UNIX timestamp

has_expired() → `bool`

Checks if the API key has expired. API Keys expire after 30 days :return: True if the key has expired, False otherwise

id

key_hash: `str`

The hash of the API key

user: `jerrycan.db.User.User`

The user associated with this API key

user_id: `int`

The ID of the user associated with this API key

verify_key (*key: str*) → bool

Checks if a given key is valid :param key: The key to check :return: True if the key is valid, False otherwise

jerrycan.db.ModelMixin module

class `jerrycan.db.ModelMixin.ModelMixin`

Bases: `object`

A mixin class that specifies a couple of methods all database models should implement

id = Column(None, Integer(), table=None, primary_key=True, nullable=False)

The ID is the primary key of the table and increments automatically

jerrycan.db.TelegramChatId module

class `jerrycan.db.TelegramChatId.TelegramChatId(*args, **kwargs)`

Bases: `jerrycan.db.ModelMixin.ModelMixin`, `sqlalchemy.ext.declarative.api.Model`

Model that describes the 'telegram_chat_ids' SQL table Maps telegram chat ids to users

__init__ (**args, **kwargs*)

Initializes the Model :param args: The constructor arguments :param kwargs: The constructor keyword arguments

chat_id: `str`

The telegram chat ID

id

send_message (*message_text: str*)

Sends a message to the telegram chat :param message_text: The message text to send :return: None

user: `jerrycan.db.User.User`

The user associated with this telegram chat ID

user_id: `int`

The ID of the user associated with this telegram chat ID

jerrycan.db.User module

class `jerrycan.db.User.User(*args, **kwargs)`

Bases: `jerrycan.db.ModelMixin.ModelMixin`, `sqlalchemy.ext.declarative.api.Model`

Model that describes the 'users' SQL table A User stores a user's information, including their email address, username and password hash

__init__ (**args, **kwargs*)

Initializes the Model :param args: The constructor arguments :param kwargs: The constructor keyword arguments

api_keys: List[ApiKey]

API keys for this user

confirmation_hash: str

The account's confirmation hash. This is the hash of a key emailed to the user. Only once the user follows the link in the email containing the key will their account be activated

confirmed: bool

The account's confirmation status. Logins should be impossible as long as this value is False.

email: str

The user's email address

get_id() → str

Method required by flask-login :return: The user's ID as a unicode string

id

property is_active

Property required by flask-login :return: True

property is_anonymous

Property required by flask-login :return: True if the user is not confirmed, False otherwise

property is_authenticated

Property required by flask-login :return: True if the user is confirmed, False otherwise

password_hash: str

The user's hashed password, salted and hashed.

telegram_chat_id: Optional[TelegramChatId]

Telegram chat ID for the user if set up

username: str

The user's username

verify_confirmation (*confirmation_key: str*) → bool

Verifies a confirmation key against the confirmation hash :param confirmation_key: The key to check :return: True if the key matches, False otherwise

verify_password (*password: str*) → bool

Verifies a password against the password hash :param password: The password to check :return: True if the password matches, False otherwise

Module contents

1.1.3 jerrycan.routes package

Subpackages

jerrycan.routes.api package

Submodules

jerrycan.routes.api.user_management module

`jerrycan.routes.api.user_management.define_blueprint` (*blueprint_name: str*) → flask.blueprints.Blueprint

Defines the blueprint for this route :param blueprint_name: The name of the blueprint :return: The blueprint

Module contents

Submodules

jerrycan.routes.decorators module

`jerrycan.routes.decorators.api` (*func: Callable*) → Callable

Decorator that handles common API patterns and ensures that the JSON response will always follow a certain pattern :param func: The function to wrap :return: The wrapper function

`jerrycan.routes.decorators.api_login_required` (*func: Callable*) → Callable

Decorator to make unauthorized API calls respond with JSON properly :param func: The function to wrap :return: The wrapped function

jerrycan.routes.static module

`jerrycan.routes.static.define_blueprint` (*blueprint_name: str*) → flask.blueprints.Blueprint

Defines the blueprint for this route :param blueprint_name: The name of the blueprint :return: The blueprint

jerrycan.routes.user_management module

`jerrycan.routes.user_management.define_blueprint` (*blueprint_name: str*) → flask.blueprints.Blueprint

Defines the blueprint for this route :param blueprint_name: The name of the blueprint :return: The blueprint

Module contents

`jerrycan.routes.blueprint_generators`: List[Tuple[Callable[[str], flask.blueprints.Blueprint], str]]

Defines the functions used to create the various blueprints as well as their names

1.1.4 jerrycan.test package

Subpackages

jerrycan.test.db package

Submodules

jerrycan.test.db.TestApiKey module

class `jerrycan.test.db.TestApiKey.TestApiKey` (*methodName='runTest'*)

Bases: `jerrycan.test.TestFramework._TestFramework`

Class that tests the ApiKey database model

test_equality()

Tests checking equality for model objects :return: None

test_expiration()

Tests if the expiration of API keys works correctly :return: None

test_hashing()
Tests using the model objects as keys in a dictionary :return: None

test_json_representation()
Tests the JSON representation of the model :return: None

test_repr()
Tests the `__repr__` method of the model class :return: None

test_string_representation()
Tests the string representation of the model :return: None

test_verifying_key()
Tests verifying an api key :return: None

jerrycan.test.db.TestModelMixin module

class jerrycan.test.db.TestModelMixin.**TestModelMixin** (*methodName='runTest'*)
Bases: jerrycan.test.TestFramework._TestFramework
Class that tests the ModelMixin class

test_enum_attributes()
Tests if enum attributes are handled correctly :return: None

test_json_representation()
Tests the JSON representation of a ModelMixin :return: None

jerrycan.test.db.TestTelegramChatId module

class jerrycan.test.db.TestTelegramChatId.**TestTelegramChatId** (*methodName='runTest'*)
Bases: jerrycan.test.TestFramework._TestFramework
Class that tests the TelegramChatId database model

test_equality()
Tests checking equality for model objects :return: None

test_hashing()
Tests using the model objects as keys in a dictionary :return: None

test_json_representation()
Tests the JSON representation of the model :return: None

test_repr()
Tests the `__repr__` method of the model class :return: None

test_sending_message()
Tests sending a message :return: None

test_string_representation()
Tests the string representation of the model :return: None

jerrycan.test.db.TestUser module

```
class jerrycan.test.db.TestUser.TestUser (methodName='runTest')
    Bases: jerrycan.test.TestFramework._TestFramework
    Class that tests the User database model

    test_equality()
        Tests checking equality for model objects :return: None

    test_flask_properties()
        Tests if the flask_login properties work as expected :return: None

    test_hashing()
        Tests using the model objects as keys in a dictionary :return: None

    test_json_representation()
        Tests the JSON representation of the model :return: None

    test_json_with_telegram_chat_id()
        Tests whether or not including children can cause recursion errors :return: None

    test_repr()
        Tests the __repr__ method of the model class :return: None

    test_string_representation()
        Tests the string representation of the model :return: None

    test_verifying_password()
        Tests verifying the password of a user :return: None
```

Module contents

jerrycan.test.misc package

Submodules

jerrycan.test.misc.TestApiCalls module

```
class jerrycan.test.misc.TestApiCalls.TestApiCalls (methodName='runTest')
    Bases: jerrycan.test.TestFramework._TestFramework
    Class that tests various API calls

    test_exception_in_api_route()
        Tests if an exception on an API route is correctly wrapped in a JSON response :return: None

    test_expired_api_key()
        Tests using an expired API key :return: None

    test_non_base64_header()
        Tests using a header that's not base64 encoded :return: None

    test_random_exception()
        Tests that the API routes catch any Exceptions without issue :return: None

    test_unauthorized_call()
        Tests and unauthorized API call :return: None
```

```
test_using_non_json_data()
    Tests sending the data as something that's not JSON :return: None
```

jerrycan.test.misc.TestConfig module

```
class jerrycan.test.misc.TestConfig.TestConfig (methodName='runTest')
    Bases: jerrycan.test.TestFramework._TestFramework

    Class that tests the config class

    test_building_base_url()
        Tests building the base URL :return: None

    test_db_config()
        Tests the database configuration :return: None

    test_dumping_environment_variables()
        Tests dumping environment variables :return: None

    test_environment_variables_definitions()
        Tests the definition of the environment variables :return: None

    test_initializing_telegram_bot_connection()
        'Tests' initializing the telegram bot connection :return: None

    test_version()
        Tests if the version is fetched correctly :return: None
```

jerrycan.test.misc.TestErrorHandling module

```
class jerrycan.test.misc.TestErrorHandling.TestErrorHandling (methodName='runTest')
    Bases: jerrycan.test.TestFramework._TestFramework

    Class that tests the flask error handling

    test_404()
        Tests if a 404 error is handled correctly :return: None

    test_api_exception()
        Tests if ApiExceptions in the API are handled correctly :return: None

    test_exception()
        Tests if unexpected exceptions are caught correctly :return: None
```

jerrycan.test.misc.TestInitialization module

```
class jerrycan.test.misc.TestInitialization.TestInitialization (methodName='runTest')
    Bases: jerrycan.test.TestFramework._TestFramework

    Tests the initialization of the flask application

    test_db_connection_error()
        Tests if a database connection error is handled correctly :return: None

    test_missing_environment_variables()
        Tests if missing environment variables are detected correctly :return: None
```

test_missing_required_template()
Tests if missing template files are detected correctly :return: None

test_no_extra_jinja()
Tests if not providing any additional jinja variables works as intended :return: None

jerrycan.test.misc.TestTelegramBackground module

class jerrycan.test.misc.TestTelegramBackground.**TestTelegramBackground** (*methodName='runTest'*)
Bases: jerrycan.test.TestFramework._TestFramework

Tests the telegram background functions

test_whoami()
Tests the whoami functionality :return: None

jerrycan.test.misc.TestWsgi module

class jerrycan.test.misc.TestWsgi.**TestWsgi** (*methodName='runTest'*)
Bases: jerrycan.test.TestFramework._TestFramework

Tests the WSGI server

test_starting_and_stopping_wsgi_server()
Tests starting and stopping the WSGI server :return: None

test_starting_background_tasks()
Tests starting background tasks :return: None

Module contents

jerrycan.test.routes package

Subpackages

jerrycan.test.routes.api package

Submodules

jerrycan.test.routes.api.TestApiKeyRoute module

class jerrycan.test.routes.api.TestApiKeyRoute.**TestApiKeyRoute** (*methodName='runTest'*)
Bases: jerrycan.test.TestFramework._TestFramework

Class that tests API-key related features

test_requesting_api_key()
Tests requesting an API key :return: None

test_requesting_invalid_api_keys()
Tests requesting API keys with invalid data :return: None

test_revoking_api_key()
Tests revoking an API key :return: None

test_unsuccessfully_revoking_api_key()
Tests unsuccessfully revoking an API key :return: None

Module contents

Submodules

jerrycan.test.routes.TestForgotRoute module

class jerrycan.test.routes.TestForgotRoute.**TestForgotRoute** (*methodName='runTest'*)
Bases: jerrycan.test.TestFramework._TestFramework

Class that tests password reset features

test_invalid_recaptcha()
Tests that invalid ReCaptcha responses are handled correctly :return: None

test_page_get()
Tests getting the page :return: None

test_resetting_password()
Tests successfully resetting a password :return: None

test_unsuccessfully_resetting_password()
Tests unsuccessfully resetting a password :return: None

jerrycan.test.routes.TestLoginRoute module

class jerrycan.test.routes.TestLoginRoute.**TestLoginRoute** (*methodName='runTest'*)
Bases: jerrycan.test.TestFramework._TestFramework

Class that tests log-in features

test_invalid_login_attempts()
Tests trying to log in with invalid credentials etc :return: None

test_logging_in_and_out()
Tests logging in successfully, then once more, then logging out :return: None

test_logging_in_with_email()
Tests logging in with an email address :return: None

test_page_get()
Tests getting the page :return: None

jerrycan.test.routes.TestProfileRoute module

class jerrycan.test.routes.TestProfileRoute.**TestProfileRoute** (*methodName='runTest'*)
Bases: jerrycan.test.TestFramework._TestFramework

Class that tests profile features

test_changing_password()
Tests changing a password :return: None

test_page_get()
Tests getting the page :return: None

```
test_unsuccessful_password_change ()  
    Tests unsuccessfully changing a password :return: None  
test_unsuccessful_user_delete ()  
    Tests unsuccessfully deleting a user :return: None  
test_user_delete ()  
    Tests deleting a user :return: None
```

jerrycan.test.routes.TestRegisterRoute module

```
class jerrycan.test.routes.TestRegisterRoute.TestRegisterRoute (methodName='runTest')  
    Bases: jerrycan.test.TestFramework._TestFramework  
  
    Class that tests registration features  
  
    test_confirming ()  
        Tests confirming a user :return: None  
  
    test_invalid_confirm ()  
        Tests invalid confirmations :return: None  
  
    test_invalid_recaptcha ()  
        Tests that invalid ReCaptcha responses are handled correctly :return: None  
  
    test_invalid_registrations ()  
        Tests registering using invalid parameters :return: None  
  
    test_page_get ()  
        Tests getting the page :return: None  
  
    test_registering_user ()  
        Tests registering a new user :return: None
```

jerrycan.test.routes.TestRegisterTelegramRoute module

```
class jerrycan.test.routes.TestRegisterTelegramRoute.TestRegisterTelegramRoute (methodName='runTest')  
    Bases: jerrycan.test.TestFramework._TestFramework  
  
    Class that tests the route used to register telegram chat IDs  
  
    test_registering ()  
        Tests registering a telegram chat ID :return: None
```

jerrycan.test.routes.TestStaticRoutes module

```
class jerrycan.test.routes.TestStaticRoutes.TestStaticRoutes (methodName='runTest')  
    Bases: jerrycan.test.TestFramework._TestFramework  
  
    Class that tests static pages  
  
    test_get_about ()  
        Tests getting the about page :return: None  
  
    test_get_index ()  
        Tests getting the index page :return: None
```



```
test_get_privacy()
    Tests getting the privacy page :return: None
```

Module contents

Submodules

jerrycan.test.TestFramework module

jerrycan.test.mocks module

Module contents

1.2 Submodules

1.3 jerrycan.Config module

```
class jerrycan.Config.Config
```

Bases: object

Class that keeps track of configuration data The class attributes should only be called after running load_config

```
API_VERSION: str = '1'
```

The API Version

```
BEHIND_PROXY: bool
```

Whether or not the site is being served by a reverse proxy like nginx.

```
DB_MODE: str
```

The database mode (for example 'sqlite' or 'mysql')

```
DB_URI: str
```

The database URI to use for database operations

```
DEBUG_LOGGING_PATH: str
```

The path to the debug logging path

```
DOMAIN_NAME: str
```

The domain name of the website

```
FLASK_SECRET: str
```

The flask secret key

```
HTTP_PORT: int
```

The port to use when serving the flask application

```
LOGGING_PATH: str
```

The path to the logging file

```
MAX_API_KEY_AGE: int = 2592000
```

The maximum age for API keys

```
MAX_USERNAME_LENGTH: int = 12
```

The maximum length of usernames

```
MIN_USERNAME_LENGTH: int = 1
```

The Minimum length for usernames

RECAPTCHA_SECRET_KEY: str

Google ReCaptcha secret key for bot detection

RECAPTCHA_SITE_KEY: str

Google ReCaptcha site key for bot detection

REQUIRED_TEMPLATES: Dict[str, str] = {'about': 'static/about.html', 'error_page': 's

Specifies required template files

SENTRY_DSN: str

The sentry DSN used for error logging

SMTP_ADDRESS: str

The SMPT Address used for sending emails

SMTP_HOST: str

The SMPT Host used for sending emails

SMTP_PASSWORD: str

The SMPT Password used for sending emails

SMTP_PORT: int

The SMPT Port used for sending emails

STRINGS: Dict[str, str] = {'401_message': 'You are not logged in', '500_message': 'T

Dictionary that defines various strings used in the application. Makes it easier to use custom phrases.

TELEGRAM_API_KEY: str

API key for a telegram bot

TELEGRAM_BOT_CONNECTION: bokkichat.connection.impl.TelegramBotConnection.TelegramBotCo

Telegram bot connection

TELEGRAM_WHOAMI: bool

Whether or not the telegram WHOAMI background thread will be started

TEMPLATE_EXTRAS: Dict[str, Callable[], Dict[str, Any]] = {'about': <function Config.

This can be used to provide the template rendering engine additional parameters, which may be necessary when adding UI elements. This is done with functions that don't expect any input and return a dictionary of keys and values to be passed to the template rendering engine

TESTING: bool

Whether or not testing is enabled

VERBOSITY: int

The verbosity level of the logging when printing to the console

VERSION: str

The current version of the application

WARNING_LOGGING_PATH: str

The path to the logging path for WARNING messages

classmethod base_url () → str

Returns The base URL of the website

classmethod dump_env_variables (path: Optional[str] = None)

Dumps all environment variables used by this application to a file :param path: The path to the file to which to dump the content.

If this is None, the file contents will be printed.

Returns None

classmethod ensure_environment_variables_present ()
 Makes sure that all required environment variables have been set. If this is not the case, the app will exit.
 :return: None

classmethod environment_variables () → Dict[str, List[str]]
 Specifies required and optional environment variables :return: The specified environment variables in two lists in
 a dictionary, grouped by whether the variables are required or optional

classmethod initialize_telegram ()
 Initializes the telegram bot connection :return: None

classmethod load_config (root_path: str, module_name: str, sentry_dsn: str)
 Loads the configuration from environment variables :param root_path: The root path of the application
 :param module_name: The name of the project's module :param sentry_dsn: The sentry DSN used for error logging :return: None

1.4 jerrycan.base module

`jerrycan.base.app = <Flask 'jerrycan.base'>`
 The Flask App

`jerrycan.base.db = <SQLAlchemy engine=None>`
 The SQLAlchemy database connection

`jerrycan.base.login_manager = <flask_login.login_manager.LoginManager object>`
 The Flask-Login Login Manager

1.5 jerrycan.enums module

class jerrycan.enums.AlertSeverity (value)
 Bases: enum.Enum

Enumeration that defines the various levels of severity an alert can have

DANGER = 'danger'
 Translates to a red alert

INFO = 'info'
 Translates to a blue alert

SUCCESS = 'success'
 Translates to a green alert

WARNING = 'warning'
 Translates to a yellow alert

1.6 jerrycan.exceptions module

exception `jerrycan.exceptions.ApiException` (*reason: str, status_code: int*)

Bases: `Exception`

Api raised when an API-related exception occurs

__init__ (*reason: str, status_code: int*)

Initializes the exception :param reason: The reason the API Exception was raised :param status_code: The status code associated with the exception

1.7 jerrycan.initialize module

`jerrycan.initialize.CREATED_BLUEPRINTS = []`

Keeps track of created blueprint names. This is necessary for unit testing with nose, because duplicate blueprint names will cause errors.

`jerrycan.initialize.init_flask` (*module_name: str, sentry_dsn: str, root_path: str, config: Type[jerrycan.Config.Config], models: List[Type[sqlalchemy.ext.declarative.api.Model]], blueprint_generators: List[Tuple[Callable[[str], flask.blueprints.Blueprint], str]], extra_jinja_vars: Optional[Dict[str, Any]] = None*)

Initializes the flask application :param module_name: The name of the module :param sentry_dsn: The sentry DSN used for error logging :param root_path: The root path of the flask application :param config: The Config class to use for configuration :param models: The database models to create :param blueprint_generators: Tuples that contain a function that generates

a blueprint and the name of the blueprint

Parameters `extra_jinja_vars` – Any extra variables to pass to jinja

Returns None

1.8 jerrycan.wsgi module

`jerrycan.wsgi.start_server` (*config: Type[jerrycan.Config.Config], task_definitions: Dict[str, Tuple[int, Callable]]*)

Starts the flask application using a cheroot WSGI server :param config: The configuration to use :param task_definitions: The background tasks, consisting of:

- the name of the task
- the time to wait before running the task again
- a function that takes no arguments that executes the task

Returns None

1.9 Module contents

CHAPTER
TWO

JERRYCAN

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

j

- jerrycan, 17
- jerrycan.background, 3
- jerrycan.background.telegram, 3
- jerrycan.base, 15
- jerrycan.Config, 13
- jerrycan.db, 5
- jerrycan.db.ApiKey, 3
- jerrycan.db.ModelMixin, 4
- jerrycan.db.TelegramChatId, 4
- jerrycan.db.User, 4
- jerrycan.enums, 15
- jerrycan.exceptions, 16
- jerrycan.initialize, 16
- jerrycan.routes, 6
- jerrycan.routes.api, 6
- jerrycan.routes.api.user_management, 5
- jerrycan.routes.decorators, 6
- jerrycan.routes.static, 6
- jerrycan.routes.user_management, 6
- jerrycan.test, 13
- jerrycan.test.db, 8
- jerrycan.test.db.TestApiKey, 6
- jerrycan.test.db.TestModelMixin, 7
- jerrycan.test.db.TestTelegramChatId, 7
- jerrycan.test.db.TestUser, 8
- jerrycan.test.misc, 10
- jerrycan.test.misc.TestApiCalls, 8
- jerrycan.test.misc.TestConfig, 9
- jerrycan.test.misc.TestErrorHandling, 9
- jerrycan.test.misc.TestInitialization, 9
- jerrycan.test.misc.TestTelegramBackground, 10
- jerrycan.test.misc.TestWsgi, 10
- jerrycan.test.mocks, 13
- jerrycan.test.routes, 13
- jerrycan.test.routes.api, 11
- jerrycan.test.routes.api.TestApiKeyRoute, 10
- jerrycan.test.routes.TestForgotRoute, 11
- jerrycan.test.routes.TestLoginRoute, 11
- jerrycan.test.routes.TestProfileRoute, 11
- jerrycan.test.routes.TestRegisterRoute, 12
- jerrycan.test.routes.TestRegisterTelegramRoute, 12
- jerrycan.test.routes.TestStaticRoutes, 12
- jerrycan.test.TestFramework, 13
- jerrycan.wsgi, 16

Symbols

`__init__()` (*jerrycan.db.ApiKey.ApiKey method*), 3

`__init__()` (*jerrycan.db.TelegramChatId.TelegramChatId method*), 4

`__init__()` (*jerrycan.db.User.User method*), 4

`__init__()` (*jerrycan.exceptions.ApiException method*), 16

A

`AlertSeverity` (*class in jerrycan.enums*), 15

`api()` (*in module jerrycan.routes.decorators*), 6

`api_keys` (*jerrycan.db.User.User attribute*), 4

`api_login_required()` (*in module jerrycan.routes.decorators*), 6

`API_VERSION` (*jerrycan.Config.Config attribute*), 13

`ApiException`, 16

`ApiKey` (*class in jerrycan.db.ApiKey*), 3

`app` (*in module jerrycan.base*), 15

B

`base_url()` (*jerrycan.Config.Config class method*), 14

`BEHIND_PROXY` (*jerrycan.Config.Config attribute*), 13

`blueprint_generators` (*in module jerrycan.routes*), 6

C

`chat_id` (*jerrycan.db.TelegramChatId.TelegramChatId attribute*), 4

`Config` (*class in jerrycan.Config*), 13

`confirmation_hash` (*jerrycan.db.User.User attribute*), 5

`confirmed` (*jerrycan.db.User.User attribute*), 5

`CREATED_BLUEPRINTS` (*in module jerrycan.initialize*), 16

`creation_time` (*jerrycan.db.ApiKey.ApiKey attribute*), 3

D

`DANGER` (*jerrycan.enums.AlertSeverity attribute*), 15

`db` (*in module jerrycan.base*), 15

`DB_MODE` (*jerrycan.Config.Config attribute*), 13

`DB_URI` (*jerrycan.Config.Config attribute*), 13

`DEBUG_LOGGING_PATH` (*jerrycan.Config.Config attribute*), 13

`define_blueprint()` (*in module jerrycan.routes.api.user_management*), 5

`define_blueprint()` (*in module jerrycan.routes.static*), 6

`define_blueprint()` (*in module jerrycan.routes.user_management*), 6

`DOMAIN_NAME` (*jerrycan.Config.Config attribute*), 13

`dump_env_variables()` (*jerrycan.Config.Config class method*), 14

E

`email` (*jerrycan.db.User.User attribute*), 5

`ensure_environment_variables_present()` (*jerrycan.Config.Config class method*), 15

`environment_variables()` (*jerrycan.Config.Config class method*), 15

F

`FLASK_SECRET` (*jerrycan.Config.Config attribute*), 13

G

`get_id()` (*jerrycan.db.User.User method*), 5

H

`has_expired()` (*jerrycan.db.ApiKey.ApiKey method*), 3

`HTTP_PORT` (*jerrycan.Config.Config attribute*), 13

I

`id` (*jerrycan.db.ApiKey.ApiKey attribute*), 3

`id` (*jerrycan.db.ModelMixin.ModelMixin attribute*), 4

`id` (*jerrycan.db.TelegramChatId.TelegramChatId attribute*), 4

`id` (*jerrycan.db.User.User attribute*), 5

`INFO` (*jerrycan.enums.AlertSeverity attribute*), 15

`init_flask()` (*in module jerrycan.initialize*), 16

`initialize_telegram()` (*jerrycan.Config.Config class method*), 15

`is_active()` (*jerrycan.db.User.User property*), 5

`is_anonymous()` (*jerrycan.db.User.User property*), 5

`is_authenticated()` (*jerrycan.db.User.User property*), 5

J

`jerrycan`
module, 17

`jerrycan.background`
module, 3

`jerrycan.background.telegram`
module, 3

`jerrycan.base`
module, 15

`jerrycan.Config`
module, 13

`jerrycan.db`
module, 5

`jerrycan.db.ApiKey`
module, 3

`jerrycan.db.ModelMixin`
module, 4

`jerrycan.db.TelegramChatId`
module, 4

`jerrycan.db.User`
module, 4

`jerrycan.enums`
module, 15

`jerrycan.exceptions`
module, 16

`jerrycan.initialize`
module, 16

`jerrycan.routes`
module, 6

`jerrycan.routes.api`
module, 6

`jerrycan.routes.api.user_management`
module, 5

`jerrycan.routes.decorators`
module, 6

`jerrycan.routes.static`
module, 6

`jerrycan.routes.user_management`
module, 6

`jerrycan.test`
module, 13

`jerrycan.test.db`
module, 8

`jerrycan.test.db.TestApiKey`
module, 6

`jerrycan.test.db.TestModelMixin`
module, 7

`jerrycan.test.db.TestTelegramChatId`
module, 7

`jerrycan.test.db.TestUser`
module, 8

`jerrycan.test.misc`
module, 10

`jerrycan.test.misc.TestApiCalls`
module, 8

`jerrycan.test.misc.TestConfig`
module, 9

`jerrycan.test.misc.TestErrorHandling`
module, 9

`jerrycan.test.misc.TestInitialization`
module, 9

`jerrycan.test.misc.TestTelegramBackground`
module, 10

`jerrycan.test.misc.TestWsgi`
module, 10

`jerrycan.test.mocks`
module, 13

`jerrycan.test.routes`
module, 13

`jerrycan.test.routes.api`
module, 11

`jerrycan.test.routes.api.TestApiKeyRoute`
module, 10

`jerrycan.test.routes.TestForgotRoute`
module, 11

`jerrycan.test.routes.TestLoginRoute`
module, 11

`jerrycan.test.routes.TestProfileRoute`
module, 11

`jerrycan.test.routes.TestRegisterRoute`
module, 12

`jerrycan.test.routes.TestRegisterTelegramRoute`
module, 12

`jerrycan.test.routes.TestStaticRoutes`
module, 12

`jerrycan.test.TestFramework`
module, 13

`jerrycan.wsgi`
module, 16

K

`key_hash` (*jerrycan.db.ApiKey.ApiKey attribute*), 3

L

`load_config()` (*jerrycan.Config.Config class method*), 15

`LOGGING_PATH` (*jerrycan.Config.Config attribute*), 13

`login_manager` (*in module jerrycan.base*), 15

M

`MAX_API_KEY_AGE` (*jerrycan.Config.Config attribute*), 13

`MAX_USERNAME_LENGTH` (*jerrycan.Config.Config attribute*), 13

- MIN_USERNAME_LENGTH (*jerrycan.Config.Config attribute*), 13
- ModelMixin (*class in jerrycan.db.ModelMixin*), 4
- module
- jerrycan, 17
 - jerrycan.background, 3
 - jerrycan.background.telegram, 3
 - jerrycan.base, 15
 - jerrycan.Config, 13
 - jerrycan.db, 5
 - jerrycan.db.ApiKey, 3
 - jerrycan.db.ModelMixin, 4
 - jerrycan.db.TelegramChatId, 4
 - jerrycan.db.User, 4
 - jerrycan.enums, 15
 - jerrycan.exceptions, 16
 - jerrycan.initialize, 16
 - jerrycan.routes, 6
 - jerrycan.routes.api, 6
 - jerrycan.routes.api.user_management, 5
 - jerrycan.routes.decorators, 6
 - jerrycan.routes.static, 6
 - jerrycan.routes.user_management, 6
 - jerrycan.test, 13
 - jerrycan.test.db, 8
 - jerrycan.test.db.TestApiKey, 6
 - jerrycan.test.db.TestModelMixin, 7
 - jerrycan.test.db.TestTelegramChatId, 7
 - jerrycan.test.db.TestUser, 8
 - jerrycan.test.misc, 10
 - jerrycan.test.misc.TestApiCalls, 8
 - jerrycan.test.misc.TestConfig, 9
 - jerrycan.test.misc.TestErrorHandling, 9
 - jerrycan.test.misc.TestInitialization, 9
 - jerrycan.test.misc.TestTelegramBackground, 10
 - jerrycan.test.misc.TestWsgi, 10
 - jerrycan.test.mocks, 13
 - jerrycan.test.routes, 13
 - jerrycan.test.routes.api, 11
 - jerrycan.test.routes.api.TestApiKeyRoute, 10
 - jerrycan.test.routes.TestForgotRoute, 11
 - jerrycan.test.routes.TestLoginRoute, 11
 - jerrycan.test.routes.TestProfileRoute, 11
 - jerrycan.test.routes.TestRegisterRoute, 12
 - jerrycan.test.routes.TestRegisterTelegramRoute, 12
 - jerrycan.test.routes.TestStaticRoutes, 12
 - jerrycan.test.TestFramework, 13
 - jerrycan.wsgi, 16
- P**
- password_hash (*jerrycan.db.User.User attribute*), 5
- R**
- RECAPTCHA_SECRET_KEY (*jerrycan.Config.Config attribute*), 13
- RECAPTCHA_SITE_KEY (*jerrycan.Config.Config attribute*), 14
- REQUIRED_TEMPLATES (*jerrycan.Config.Config attribute*), 14
- S**
- send_message () (*jerrycan.db.TelegramChatId.TelegramChatId method*), 4
- SENTRY_DSN (*jerrycan.Config.Config attribute*), 14
- SMTP_ADDRESS (*jerrycan.Config.Config attribute*), 14
- SMTP_HOST (*jerrycan.Config.Config attribute*), 14
- SMTP_PASSWORD (*jerrycan.Config.Config attribute*), 14
- SMTP_PORT (*jerrycan.Config.Config attribute*), 14
- start_server () (*in module jerrycan.wsgi*), 16
- STRINGS (*jerrycan.Config.Config attribute*), 14
- SUCCESS (*jerrycan.enums.AlertSeverity attribute*), 15
- T**
- TELEGRAM_API_KEY (*jerrycan.Config.Config attribute*), 14
- TELEGRAM_BOT_CONNECTION (*jerrycan.Config.Config attribute*), 14
- telegram_chat_id (*jerrycan.db.User.User attribute*), 5
- TELEGRAM_WHOAMI (*jerrycan.Config.Config attribute*), 14
- telegram_whoami () (*in module jerrycan.background.telegram*), 3
- TelegramChatId (*class in jerrycan.db.TelegramChatId*), 4
- TEMPLATE_EXTRAS (*jerrycan.Config.Config attribute*), 14
- test_404 () (*jerrycan.test.misc.TestErrorHandling.TestErrorHandling method*), 9
- test_api_exception () (*jerrycan.test.misc.TestErrorHandling.TestErrorHandling method*), 9
- test_building_base_url () (*jerrycan.test.misc.TestConfig.TestConfig method*), 9

test_changing_password()	(jerrycan.test.routes.TestProfileRoute.TestProfileRoute method), 11	test_hashing()	(jerrycan.test.db.TestApiKey.TestApiKey method), 6
test_confirming()	(jerrycan.test.routes.TestRegisterRoute.TestRegisterRoute method), 12	test_hashing()	(jerrycan.test.db.TestTelegramChatId.TestTelegramChatId method), 7
test_db_config()	(jerrycan.test.misc.TestConfig.TestConfig method), 9	test_hashing()	(jerrycan.test.db.TestUser.TestUser method), 8
test_db_connection_error()	(jerrycan.test.misc.TestInitialization.TestInitialization method), 9	test_initializing_telegram_bot_connection()	(jerrycan.test.misc.TestConfig.TestConfig method), 9
test_dumping_environment_variables()	(jerrycan.test.misc.TestConfig.TestConfig method), 9	test_invalid_confirm()	(jerrycan.test.routes.TestRegisterRoute.TestRegisterRoute method), 12
test_enum_attributes()	(jerrycan.test.db.TestModelMixin.TestModelMixin method), 7	test_invalid_login_attempts()	(jerrycan.test.routes.TestLoginRoute.TestLoginRoute method), 11
test_environment_variables_definitions()	(jerrycan.test.misc.TestConfig.TestConfig method), 9	test_invalid_recaptcha()	(jerrycan.test.routes.TestForgotRoute.TestForgotRoute method), 11
test_equality()	(jerrycan.test.db.TestApiKey.TestApiKey method), 6	test_invalid_recaptcha()	(jerrycan.test.routes.TestRegisterRoute.TestRegisterRoute method), 12
test_equality()	(jerrycan.test.db.TestTelegramChatId.TestTelegramChatId method), 7	test_invalid_registrations()	(jerrycan.test.routes.TestRegisterRoute.TestRegisterRoute method), 12
test_equality()	(jerrycan.test.db.TestUser.TestUser method), 8	test_json_representation()	(jerrycan.test.db.TestApiKey.TestApiKey method), 7
test_exception()	(jerrycan.test.misc.TestErrorHandling.TestErrorHandling method), 9	test_json_representation()	(jerrycan.test.db.TestModelMixin.TestModelMixin method), 7
test_exception_in_api_route()	(jerrycan.test.misc.TestApiCalls.TestApiCalls method), 8	test_json_representation()	(jerrycan.test.db.TestTelegramChatId.TestTelegramChatId method), 7
test_expiration()	(jerrycan.test.db.TestApiKey.TestApiKey method), 6	test_json_representation()	(jerrycan.test.db.TestUser.TestUser method), 8
test_expired_api_key()	(jerrycan.test.misc.TestApiCalls.TestApiCalls method), 8	test_json_with_telegram_chat_id()	(jerrycan.test.db.TestUser.TestUser method), 8
test_flask_properties()	(jerrycan.test.db.TestUser.TestUser method), 8	test_logging_in_and_out()	(jerrycan.test.routes.TestLoginRoute.TestLoginRoute method), 11
test_get_about()	(jerrycan.test.routes.TestStaticRoutes.TestStaticRoutes method), 12	test_logging_in_with_email()	(jerrycan.test.routes.TestLoginRoute.TestLoginRoute method), 11
test_get_index()	(jerrycan.test.routes.TestStaticRoutes.TestStaticRoutes method), 12	test_missing_environment_variables()	(jerrycan.test.misc.TestInitialization.TestInitialization method), 9
test_get_privacy()	(jerrycan.test.routes.TestStaticRoutes.TestStaticRoutes method), 12	test_missing_required_template()	(jerrycan.test.misc.TestInitialization.TestInitialization method), 9
		test_no_extra_jinja()	(jerrycan.test.misc.TestInitialization.TestInitialization method), 9

rycan.test.misc.TestInitialization.TestInitialization method), 10
rycan.test.db.TestApiKey.TestApiKey method), 7
test_non_base64_header() (*rycan.test.misc.TestApiCalls.TestApiCalls* method), 8
test_string_representation() (*rycan.test.db.TestTelegramChatId.TestTelegramChatId* method), 7
test_page_get() (*rycan.test.routes.TestForgotRoute.TestForgotRoute* method), 11
test_string_representation() (*rycan.test.db.TestUser.TestUser* method), 8
test_page_get() (*rycan.test.routes.TestLoginRoute.TestLoginRoute* method), 11
test_unauthorized_call() (*rycan.test.misc.TestApiCalls.TestApiCalls* method), 8
test_page_get() (*rycan.test.routes.TestProfileRoute.TestProfileRoute* method), 11
test_unsuccessful_password_change() (*rycan.test.routes.TestProfileRoute.TestProfileRoute* method), 11
test_page_get() (*rycan.test.routes.TestRegisterRoute.TestRegisterRoute* method), 12
test_unsuccessful_user_delete() (*rycan.test.routes.TestProfileRoute.TestProfileRoute* method), 12
test_random_exception() (*rycan.test.misc.TestApiCalls.TestApiCalls* method), 8
test_unsuccessfully_resetting_password() (*jerrycan.test.routes.TestForgotRoute.TestForgotRoute* method), 11
test_registering() (*rycan.test.routes.TestRegisterTelegramRoute.TestRegisterTelegramRoute* method), 12
test_unsuccessfully_revoking_api_key() (*jerrycan.test.routes.api.TestApiKeyRoute.TestApiKeyRoute* method), 10
test_registering_user() (*rycan.test.routes.TestRegisterRoute.TestRegisterRoute* method), 12
test_user_delete() (*rycan.test.routes.TestProfileRoute.TestProfileRoute* method), 12
test_repr() (*jerrycan.test.db.TestApiKey.TestApiKey* method), 7
test_using_non_json_data() (*rycan.test.misc.TestApiCalls.TestApiCalls* method), 8
test_repr() (*jerrycan.test.db.TestTelegramChatId.TestTelegramChatId* method), 7
test_verifying_key() (*rycan.test.db.TestApiKey.TestApiKey* method), 7
test_repr() (*jerrycan.test.db.TestUser.TestUser* method), 8
test_verifying_password() (*rycan.test.db.TestUser.TestUser* method), 8
test_requesting_api_key() (*rycan.test.routes.api.TestApiKeyRoute.TestApiKeyRoute* method), 10
test_verifying_password() (*rycan.test.db.TestUser.TestUser* method), 8
test_requesting_invalid_api_keys() (*rycan.test.routes.api.TestApiKeyRoute.TestApiKeyRoute* method), 10
test_version() (*rycan.test.misc.TestConfig.TestConfig* method), 9
test_resetting_password() (*rycan.test.routes.TestForgotRoute.TestForgotRoute* method), 11
test_whoami() (*rycan.test.misc.TestTelegramBackground.TestTelegramBackground* method), 10
test_revoking_api_key() (*rycan.test.routes.api.TestApiKeyRoute.TestApiKeyRoute* method), 10
TestApiCalls (class in *rycan.test.misc.TestApiCalls*), 8
TestApiKey (class in *jerrycan.test.db.TestApiKey*), 6
test_sending_message() (*rycan.test.db.TestTelegramChatId.TestTelegramChatId* method), 7
TestApiKeyRoute (class in *rycan.test.routes.api.TestApiKeyRoute*), 10
TestConfig (class in *jerrycan.test.misc.TestConfig*), 9
test_starting_and_stopping_wsgi_server() (*jerrycan.test.misc.TestWsgi.TestWsgi* method), 10
TestErrorHandling (class in *rycan.test.misc.TestErrorHandling*), 9
TestForgotRoute (class in *rycan.test.routes.TestForgotRoute*), 11
test_starting_background_tasks() (*rycan.test.misc.TestWsgi.TestWsgi* method), 10
TESTING (*jerrycan.Config.Config* attribute), 14
TestInitialization (class in *rycan.test.misc.TestInitialization*), 9

TestLoginRoute (class in *jerrycan.test.routes.TestLoginRoute*), 11
TestModelMixin (class in *jerrycan.test.db.TestModelMixin*), 7
TestProfileRoute (class in *jerrycan.test.routes.TestProfileRoute*), 11
TestRegisterRoute (class in *jerrycan.test.routes.TestRegisterRoute*), 12
TestRegisterTelegramRoute (class in *jerrycan.test.routes.TestRegisterTelegramRoute*), 12
TestStaticRoutes (class in *jerrycan.test.routes.TestStaticRoutes*), 12
TestTelegramBackground (class in *jerrycan.test.misc.TestTelegramBackground*), 10
TestTelegramChatId (class in *jerrycan.test.db.TestTelegramChatId*), 7
TestUser (class in *jerrycan.test.db.TestUser*), 8
TestWsgi (class in *jerrycan.test.misc.TestWsgi*), 10

U

User (class in *jerrycan.db.User*), 4
user (*jerrycan.db.ApiKey.ApiKey* attribute), 3
user (*jerrycan.db.TelegramChatId.TelegramChatId* attribute), 4
user_id (*jerrycan.db.ApiKey.ApiKey* attribute), 4
user_id (*jerrycan.db.TelegramChatId.TelegramChatId* attribute), 4
username (*jerrycan.db.User.User* attribute), 5

V

VERBOSEITY (*jerrycan.Config.Config* attribute), 14
verify_confirmation() (*jerrycan.db.User.User* method), 5
verify_key() (*jerrycan.db.ApiKey.ApiKey* method), 4
verify_password() (*jerrycan.db.User.User* method), 5
VERSION (*jerrycan.Config.Config* attribute), 14

W

WARNING (*jerrycan.enums.AlertSeverity* attribute), 15
WARNING_LOGGING_PATH (*jerrycan.Config.Config* attribute), 14