msaSDK Module

`.service`

Main Service Module for MSAApp.

Initialize with a MSAServiceDefintion Instance to control the features and functions of the MSAApp.

Attributes

password_helper `module-attribute`

```
password_helper = PasswordHelper(security_context)
```

> Password Helper Instance

security `module-attribute`

```
security = getMSASecurity()
```

> MSASecurity instance

security_context `module-attribute`

```
security_context = CryptContext(
    schemes=["bcrypt"], deprecated="auto"
)
```

> Security Context for Password Helper

Classes

MSAApp

> Bases: `MSAFastAPI`

Creates an application msaSDK instance.

> ✏️ **Note** ∨
>
> As with FastApi the MSAApp provides two events: `startup` : A list of callables to run on application startup. Startup handler callables do not take any arguments, and may be be either standard functions, or async functions. `shutdown` : A list of callables to run on application shutdown. Shutdown handler callables do not take any arguments, and may be be either standard functions, or async functions. Those are also used internally, which are triggered before the external events.
>
> Do not include the `self` parameter in the `Args` section.

PARAMETER	DESCRIPTION
`settings`	MSAServiceDefinition (Must be provided), instance of a service definition with all settings **TYPE:** `MSAServiceDefinition`
`timers`	MSATimers instance Default None, provide a MSATimers instance and it will start the scheduler internaly **TYPE:** `MSATimers` **DEFAULT:** `None`
`sql_models`	List of SQLModel Default None, provide list of your SQLModel Classes and the instance can create CRUD API and if site is enabled also UI for CRUD **TYPE:** `List[SQLModel]` **DEFAULT:** `None`
`auto_mount_site`	Default True, if site is enabled in settings and this is true, mounts the site in internal startup event. **TYPE:** `bool` **DEFAULT:** `True`

ATTRIBUTE	DESCRIPTION
`logger`	loguru logger instance
`auto_mount_site`	bool auto_mount_site **TYPE:** `bool`
`settings`	MSAServiceDefinition settings instance.
`timers`	MSATimers = timers **TYPE:** `MSATimers`

ATTRIBUTE	DESCRIPTION
`healthdefinition`	MSAHealthDefinition settings.healthdefinition **TYPE:** `MSAHealthDefinition`
`limiter`	Limiter = None **TYPE:** `Limiter`
`db_engine`	AsyncEngine = Db Engine instance **TYPE:** `AsyncEngine`
`sql_models`	List[SQLModel] = sql_models **TYPE:** `List[SQLModel]`
`sql_cruds`	List[MSASQLModelCrud] = [] **TYPE:** `List[MSASQLModelCrud]`
`scheduler`	MSAScheduler = None **TYPE:** `MSAScheduler`
`site`	AdminSite Admin/Auth Site instance.
`scheduler_task`	The Task instance that runs the Scheduler in the Background **TYPE:** `Task`
`ROOTPATH`	str os.path.join(os.path.dirname(**file**))

Attributes

Base `instance-attribute`

```
Base: DeclarativeMeta = declarative_base()
```

ROOTPATH `instance-attribute`

```
ROOTPATH = os.path.join(os.path.dirname(__file__))
```

auto_mount_site `instance-attribute`

```
auto_mount_site: bool = auto_mount_site
```

db_engine `instance-attribute`

```
db_engine: AsyncEngine = None
```

graphql_app `instance-attribute`

```
graphql_app: GraphQLRouter = None
```

graphql_schema `instance-attribute`

```
graphql_schema: schema = None
```

healthcheck `instance-attribute`

```
healthcheck: health.MSAHealthCheck = None
```

healthdefinition `instance-attribute`

```
healthdefinition: MSAHealthDefinition = (
    self.settings.healthdefinition
)
```

limiter `instance-attribute`

```
limiter: Limiter = None
```

logger `instance-attribute`

```
logger = logger
```

scheduler `instance-attribute`

```
scheduler: MSAScheduler = None
```

scheduler_task `instance-attribute`

```
scheduler_task: Task = None
```

settings `instance-attribute`

```
settings = settings
```

site `instance-attribute`

```
site = None
```

sql_cruds `instance-attribute`

```
sql_cruds: List[MSASQLModelCrud] = []
```

sql_models `instance-attribute`

```
sql_models: List[SQLModel] = sql_models
```

templates `instance-attribute`

```
templates = Jinja2Templates(
    directory=self.settings.templates_dir
)
```

timers `instance-attribute`

```
timers: MSATimers = timers
```

Functions

__init__

```
__init__(
    settings: MSAServiceDefinition,
    timers: MSATimers = None,
    sql_models: List[SQLModel] = None,
    auto_mount_site: bool = True,
    *args,
    **kwargs
) -> None
```

get_healthcheck `async`

```
get_healthcheck(request: Request) -> ORJSONResponse
```

Get Healthcheck Status

get_scheduler_status `async`

```
get_scheduler_status(
    request: Request,
) -> MSASchedulerStatus
```

Get Service Status Info

get_services_definition

```
get_services_definition(
    request: Request,
) -> MSAServiceDefinition
```

Get Service Definition Info

get_services_openapi_info

```
get_services_openapi_info(
    request: Request,
) -> MSAOpenAPIInfo
```

Get Service OpenAPI Info

get_services_openapi_schema

```
get_services_openapi_schema(
    request: Request,
) -> ORJSONResponse
```

Get Service OpenAPI Schema

get_services_settings

```
get_services_settings(request: Request) -> ORJSONResponse
```

Get Service OpenAPI Schema

get_services_status `async`

```
get_services_status(request: Request) -> MSAServiceStatus
```

Get Service Status Info

index_page

```
index_page(request: Request) -> _TemplateResponse
```

Get Service Index.html Page

monitor `async`

```
monitor(request: Request) -> _TemplateResponse
```

Simple Service Monitor Page. Only works if pages is enabled in MSAServiceDefinition :param request: :return:

monitor_inline `async`

```
monitor_inline(request: Request) -> _TemplateResponse
```

Simple Monitor Page as Inline without head and body tags. Only works if pages is enabled in MSAServiceDefinition :param request: :return:

mount_site

```
mount_site() -> None
```

msa_exception_handler `async`

```
msa_exception_handler(request: Request, exc: HTTPException)
```

Handles all HTTPExceptions if enabled with HTML Response or forward error if the code is in the exclude settings list. :param request: :type request: :param exc: :type exc: :return: :rtype:

msa_exception_handler_disabled `async`

```
msa_exception_handler_disabled(
    request: Request, exc: HTTPException
) -> JSONResponse
```

Handles all HTTPExceptions if Disabled with JSON Response. :param request: :type request: :param exc: :type exc: :return: :rtype:

profiler

```
profiler(request: Request) -> _TemplateResponse
```

Simple Profiler Page. Only works if pages is enabled in MSAServiceDefinition :param request:
:return:

shutdown_event `async`

```
shutdown_event() -> None
```

startup_event `async`

```
startup_event() -> None
```

:return: :rtype:

testpage

```
testpage(request: Request) -> _TemplateResponse
```

Simple Testpage to see if the Micro Service is up and running. Only works if pages is enabled in
MSAServiceDefinition :param request: :return:

validation_exception_handler `async`

```
validation_exception_handler(
    request: Request, exc: RequestValidationError
) -> JSONResponse
```

MSAOpenAPIInfo

Bases: `SQLModel`

MSAOpenAPIInfo Pydantic Response Class

Attributes

name `class-attribute`

```
name: str = 'msaSDK Service'
```

Service Name.

tags `class-attribute`

```
tags: Optional[List[str]] = None
```

OpenAPI Tags.

url `class-attribute`

```
url: str = '/openapi.json'
```

OpenAPI URL.

version `class-attribute`

```
version: str = '0.0.0'
```

API Version.

MSASchedulerStatus

Bases: `SQLModel`

MSASchedulerStatus Pydantic Response Class

Attributes

message `class-attribute`

```
message: Optional[str] = 'None'
```

Optional Message Text

name `class-attribute`

```
name: Optional[str] = 'msaSDK Service'
```

Service Name.

timers `class-attribute`

```
timers: Optional[List[MSATimerStatus]] = []
```

Optional MSATimerStatus List

MSAServiceStatus

Bases: `SQLModel`

MSAServiceStatus Pydantic Response Class

Attributes

healthy `class-attribute`

```
healthy: Optional[str] = 'None'
```

Health status

message `class-attribute`

```
message: Optional[str] = 'None'
```

Optional Message Text

name `class-attribute`

```
name: Optional[str] = 'msaSDK Service'
```

Service Name.

MSATimerStatus

Bases: `SQLModel`

MSATimerStatus Pydantic Response Class

Attributes

func `class-attribute`

```
func: Optional[str] = None
```

Timer Handler Function.

mark_HH_MM `class-attribute`

```
mark_HH_MM: Optional[str] = None
```

Mark for Schedule

mode `class-attribute`

```
mode: Optional[str] = None
```

Timer Mode.

Functions

getSecretKey

```
getSecretKey()
```

Get Secret Key for Token creation from OS Environment Variable **SECRET_KEY_TOKEN**

RETURNS	DESCRIPTION
`key`	The SECRET_KEY_TOKEN.

getSecretKeyCSRF

```
getSecretKeyCSRF() -> str
```

Get Secret Key for CSRF Middleware from OS Environment Variable **SECRET_KEY_CSRF**

RETURNS	DESCRIPTION
`key`	The SECRET_KEY_CSRF. **TYPE:** `str`

getSecretKeySessions

```
getSecretKeySessions()
```

Get Secret Key for Session Middleware from OS Environment Variable **SECRET_KEY_SESSIONS**

RETURNS	DESCRIPTION
`key`	The SECRET_KEY_SESSIONS.

Last update: September 13, 2022

Created: September 13, 2022