msaSDK Module

`.db.crud.base`

Classes

MSABaseCrud

Bases: `MSARouterMixin`

Register Router Base Class

> ✏️ **Note**   ⌄
>
> Database tables have and can only have one self-adding primary key.

Attributes

list_per_page_max `class-attribute`

```
list_per_page_max: int = None
```

> Maximum number of data per page to read in batches. Default: None, no limit.

paginator `instance-attribute`

```
paginator: Type[MSACRUDPaginator] = MSACRUDPaginator
```

pk_name `class-attribute`

```
pk_name: str = 'id'
```

> Current model primary key string, default: id.
>
> Note: Database tables have and can only have one self-adding primary key.

router `instance-attribute`

```
router = router
```

schema_create `class-attribute`

```
schema_create: Type[BaseModel] = None
```

> Create data submission protocol, default: self.schema_model

schema_filter `class-attribute`

```
schema_filter: Type[BaseModel] = None
```

> Batch query data filter submission protocol, default: self.schema_model

schema_list `class-attribute`

```
schema_list: Type[BaseModel] = None
```

> Batch query data return protocol, default: self.schema_model

schema_model `instance-attribute`

```
schema_model = schema_model or self.schema_model
```

schema_read `class-attribute`

```
schema_read: Type[BaseModel] = None
```

> Read data return protocol, default: self.schema_model

schema_update `class-attribute`

```
schema_update: Type[BaseModel] = None
```

> Update data submission protocol, default: self.schema_model

Functions

__init__

```
__init__(
    schema_model: Type[BaseModel], router: APIRouter = None
)
```

error_data_handle

```
error_data_handle(request: Request)
```

error_execute_sql

```
error_execute_sql(request: Request, error: Exception)
```

has_create_permission `async`

```
has_create_permission(
    request: Request, obj: Optional[BaseModel], **kwargs
) -> bool
```

> Check for single create permission.

has_delete_permission `async`

```
has_delete_permission(
    request: Request, item_id: Optional[List[str]], **kwargs
) -> bool
```

> Check for single delete permission.

has_list_permission `async`

```
has_list_permission(
    request: Request,
    paginator: Optional[MSACRUDPaginator],
    filters: Optional[BaseModel],
    **kwargs
) -> bool
```

> Check for bulk query permission.

has_read_permission `async`

```
has_read_permission(
    request: Request, item_id: Optional[List[str]], **kwargs
) -> bool
```

> Check for single read permission.

has_update_permission `async`

```
has_update_permission(
    request: Request,
    item_id: Optional[List[str]],
    obj: Optional[BaseModel],
    **kwargs
) -> bool
```

Check for single update permission.

register_crud

```
register_crud(
    schema_list: Type[BaseModel] = None,
    schema_filter: Type[BaseModel] = None,
    schema_create: Type[BaseModel] = None,
    schema_read: Type[BaseModel] = None,
    schema_update: Type[BaseModel] = None,
    list_max_per_page: int = None,
    depends_list: List[Depends] = None,
    depends_read: List[Depends] = None,
    depends_create: List[Depends] = None,
    depends_update: List[Depends] = None,
    depends_delete: List[Depends] = None,
) -> MSABaseCrud
```

Register Crud routing.

route_create `property`

```
route_create() -> Callable[..., Any]
```

Single/batch creation of routing functions. Support for synchronous/asynchronous functions.

route_delete `property`

```
route_delete() -> Callable[..., Any]
```

Single/batch delete routing functions. Support for synchronous/asynchronous functions.

route_list `property`

```
route_list() -> Callable[..., Any]
```

Batch read routing functions. Support for synchronous/asynchronous functions.

route_read `property`

```
route_read() -> Callable[..., Any]
```

Single/batch read routing functions. Support for synchronous/asynchronous functions.

route_update `property`

```
route_update() -> Callable[..., Any]
```

Single/batch update routing functions. Support for synchronous/asynchronous functions.

router_prefix `property`

```
router_prefix()
```

schema_name_prefix `property`

```
schema_name_prefix()
```

MSARouterMixin

Register Router

Attributes

router `instance-attribute`

```
router = self.get_router()
```

router_permission_depend `class-attribute`

```
router_permission_depend: Callable = None
```

router_prefix `class-attribute`

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

Functions

__init__

```
__init__()
```

error_no_router_permission

```
error_no_router_permission(request: Request)
```

get_router

```
get_router() -> APIRouter
```

Functions

Last update: September 13, 2022

Created: September 13, 2022