

# Podlaski Urząd Wojewódzki w Białymstoku

## Dokumentacja EZD API

---

PODSTAWOWA KONFIGURACJA ORAZ OPIS INTERFEJSÓW

## 1 Spis treści

2	Cel dokumentu .....	5
2.1	Zastosowane skróty i pojęcia .....	5
3	Podstawowe informacje na temat silnika API .....	7
4.1	W języku c#/.net.....	8
4.1.2	Przykłady komunikacji .....	16
5	Opis interfejsów oraz ich wykorzystanie .....	18
5.1	Pobieranie całego drzewa Rwa z konkretnego rocznika .....	18
5.1.1	DTO request – PobierzRwaPoRocznikuRequest .....	18
5.1.2	DTO response – PobierzRwaPoRocznikuResponse .....	18
5.2	Pobieranie wszystkich jednostek w strukturze .....	19
5.3	DTO request – PobierzWszytkieJednostkiRequest.....	19
5.4	DTO response – PobierzWszytkieJednostkiResponse .....	19
5.5	Pobieranie załączników z storage.....	19
5.5.1	DTO request – PobierzZalacznikRequest.....	20
5.5.2	DTO response – PobierzZalacznikResponse .....	20
5.6	Przenoszenie pracownika w strukturze.....	21
5.6.1	DTO request – PrzeniesPracownikaRequest .....	21
5.6.2	DTO response – PrzeniesPracownikaResponse.....	21
5.7	Dodaj Pracownika.....	22
5.7.1	DTO request – DodajPracownikaRequest .....	22
5.7.2	DTO response – DodajPracownikaResponse.....	23
5.8	Pobierz pracowników należących do jednostki niezależnie od typu jednostki.....	24
5.8.1	DTO request – PobierzPracownikowNalezacychDoJednostkiRequest.....	24
5.8.2	DTO response – PobierzPracownikowNalezacychDoJednostkiResponse .....	24
5.9	Pobierz Pracownika .....	24
5.9.1	DTO request – PobierzPracownikaReq.....	25
5.9.2	DTO response – PobierzPracownikaRes.....	27
5.10	Pobierz wszystkie identyfikatory dokumentów znajdujące się w koszulce .....	29
5.10.1	DTO request - PobierzIdentyfikatoryDokumentowKoszulkiRequest .....	29
5.10.2	DTO response - PobierzIdentyfikatoryDokumentowKoszulkiResponse.....	29

5.11	Dodawanie załączników .....	29
5.11.1	DTO request – DodajZalacznikRequest .....	29
5.11.2	DTO response – DodajZalacznikRresponse .....	30
5.12	Aktualizacja informacji o dokumencie .....	30
5.12.1	DTO request – AktualizujDokumentReq.....	31
5.12.2	DTO response – AktualizujDokumentRes.....	33
5.13	Pobieranie identyfikatorów koszulek których właścicielem jest określony Pracownik. ....	33
5.13.1	DTO request – PobierzIdentyfikatoryKoszulekRequest.....	33
5.13.2	DTO response – PobierzIdentyfikatoryKoszulekResponse .....	33
5.14	Aktualizacja danych koszulki .....	33
5.14.1	DTO request – AktualizujKoszulkeRequest.....	37
5.14.2	DTO response – AktualizujKoszulkeResponse .....	38
5.15	PobierzWszytskichPracownikow .....	39
5.15.1	DTO request – PobierzWszytskichPracownikowRequest.....	39
5.15.2	DTO response – PobierzWszytskichPracownikowResponse .....	39
5.16	PrzekazKoszulke.....	39
5.16.1	DTO request – PrzekazKoszulkeReq .....	39
5.16.2	DTO reponse – PrzekazKoszulkeRes .....	41
5.17	RejestrujSprawe .....	41
5.17.1	DTO request – RejestrujSpraweReq .....	41
5.17.2	DTO response – RejestrujSpraweRes .....	43
5.18	UtworzKoszulke .....	43
5.18.1	DTO request – UtworzKoszulkeReq.....	44
5.18.2	DTO response – UtworzKoszulkeRes.....	45
5.19	ZakonczKoszulke.....	46
5.19.1	DTO request – ZakonczKoszulkeReq .....	46
5.19.2	DTO response – ZakonczKoszulkeRes.....	47
5.20	Pobieranie jednostki po identyfikatorze .....	47
5.20.1	DTO request – PobierzJednostkęPoldRequest .....	47
5.20.2	DTO response – PobierzJednostkęPoldResponse .....	47
5.21	Pobieranie koszulki po identyfikatorze koszulki.....	47
5.21.1	DTO request – PobierzKoszulkePoldRequest .....	48
5.21.2	DTO response – PobierzKoszulkePoldResponse .....	48

5.22	Pobierz koszulkę po znaku sprawy .....	48
5.22.1	DTO request – PobierzKoszulkePoZnakuSprawyRequest.....	48
5.22.2	DTO response – PobierzKoszulkePoZnakuSprawyResponse .....	48
6	Globalne Data Transfer Object .....	48
6.1	PismoDto (Koszulka).....	48
6.2	StanowiskoDto (Stanowisko pracownika) .....	49
6.3	PracownikDto .....	51
6.4	JednostkaDto.....	54
6.5	TeczkaRwaDto .....	55
6.6	DokumentTypeDto .....	55
6.7	Enumeratory.....	56
6.7.1	RodzajKoszulki .....	56

## 2 Cel dokumentu

Dokumentacja EZD PUW API ma na celu umożliwienie zapoznania się przez integratorów z strukturą oraz przykładami konsumpcji interfejsów napisanych autorstwa Podlaskiego Urzędu Wojewódzkiego

### 2.1 Zastosowane skróty i pojęcia

W poniższej tabeli przedstawiony został słownik pojęć i skrótów stosowanych w niniejszym dokumencie.

Pojęcie/skrót	Wyjaśnienie
EZD PUW	System EZD autorstwa Podlaskiego Urzędu Wojewódzkiego w Białymstoku, system będący własnością Skarbu Państwa, przedmiot Projektu – system, który będzie rozwijany i wzmacniany jako jednolite narzędzie administracji rządowej RP w ramach dostarczonych produktów przedmiotowego Projektu.
API	Interfejs programistyczny aplikacji (ang. Application Programming Interface, API) – zbiór metod reguł i obiektów wykorzystywanych do komunikacji
Interfejs (ang. interface)	definicja abstrakcyjnego typu posiadającego jedynie operacje, a nie dane. Kiedy w konkretnej klasie zdefiniowane są wszystkie metody interfejsu mówimy, że klasa implementuje dany interfejs
Dto (Data Transfer Object)	Służy do przenoszenia danych z warstwy danych do obiektu biznesowego.
Route	To register your custom REST URLs, you can use the Route attribute on the request
IReturn< Nazwa klasy >	Jaki obiekt zostanie zwrócony po wykonaniu poprawnego żądania <Nazwa klasy>
REST	RESTful Webservices (inaczej RESTful web API) jest usługą sieciową zaimplementowaną na bazie protokołu HTTP i głównych zasad wzorca REST.

Pojęcie/skrót	Wyjaśnienie
ENUM / ENUMERATOR	Typ wyliczeniowy – rodzaj typu danych zawierający listę wartości reprezentowanych za pomocą literałów wyliczeniowych, jakie może przyjmować zmienna tego typu. Typ wyliczeniowy pełni nieocenioną funkcję w metaprogramowaniu, gdyż pozwala na tworzenie stałych w chwili kompilacji.
Exception /Wyjątek	jest mechanizmem przepływu sterowania używanym w mikroprocesorach oraz współczesnych językach programowania do obsługi zdarzeń wyjątkowych, a w szczególności błędów, których wystąpienie zmienia prawidłowy przebieg wykonywania programu. W momencie zajścia niespodziewanego zdarzenia generowany jest wyjątek, który musi zostać obsłużony poprzez zapamiętanie bieżącego stanu programu i przejście do procedury jego obsługi. W niektórych sytuacjach po obsłużeniu wyjątku można powrócić do wykonywania przerwanego kodu, korzystając z zapamiętanych informacji stanu. Przykładowo obsługa błędu braku strony pamięci polega najczęściej na pobraniu brakującej strony z pliku wymiany, co umożliwia kontynuowanie pracy programu, natomiast błąd dzielenia przez zero powoduje, że wykonywanie dalszych obliczeń nie ma sensu i musi zostać definitywnie przerwane.

### 3 Podstawowe informacje na temat silnika API

Wykorzystywanym silnikiem do zaimplementowania API jest ServiceStack. Dokumentacja dotycząca integracji z serwisie:

<https://github.com/ServiceStack/ServiceStack/wiki/C%23-client>

Dokumentacja dotycząca całego serwisu i integracji z innymi językami programowania :

<https://github.com/ServiceStack/ServiceStack/wiki>

## 4 Przykłady konfiguracji ,połączeń i komunikacji z API

### 4.1 W języku c#/.net

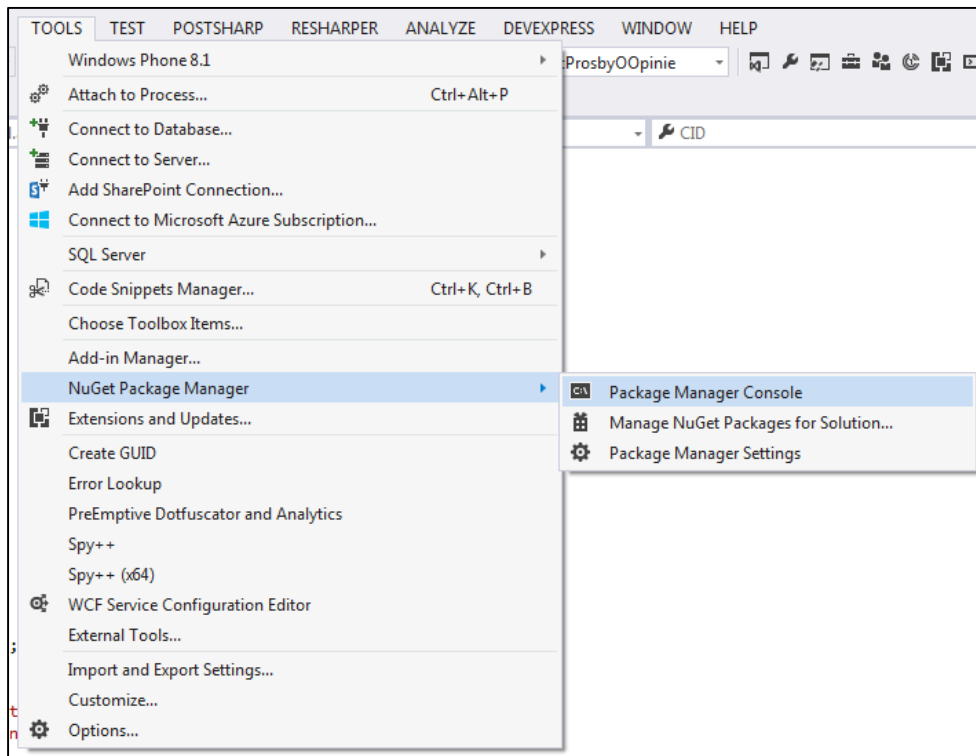
Wymagania podstawowe do komunikacji z serwisem integracji:

- Microsoft Visual studio w wersjach minimum 2012 i wyżej
- Pakiety NuGet „ServiceStack.Client”, „ServiceStack.HttpClient”
- ServiceStackVS Extension do Visual studio (wymagane do auto dodawania referencji ServiceStack).



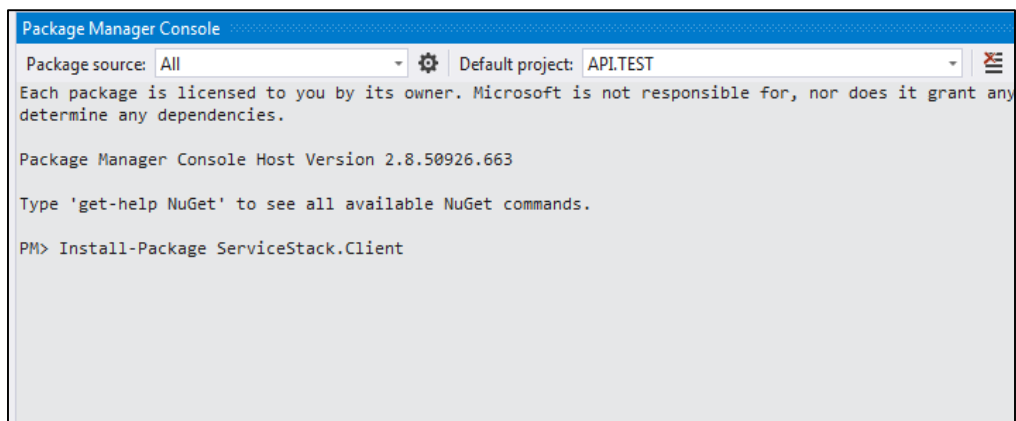
#### 4.1.1.1 Instalacja pakietów ServiceStack poprzez konsolę „Package Manager Console”

1. Uruchomienie konsoli zarządzania pakietami NuGet



Rysunek 1 Uruchomienie konsoli zarządzania pakietami NuGet vs2013

2. Instalacja wymaganych pakietów za pomocą komendy `Install-Package ServiceStack.Client`.

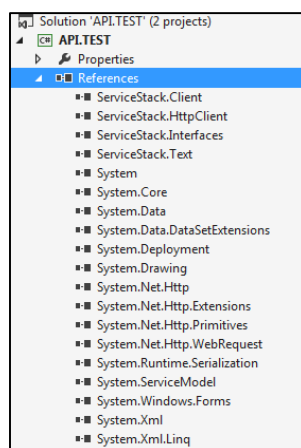


Rysunek 2 Instalowanie pakietu ServiceStack.Client

```
Package Manager Console
Package source: nuget.org Default project: API.TEST
Successfully removed 'ServiceStack.Interfaces 4.0.48' from API.TEST.
Adding 'ServiceStack.Interfaces 4.0.50' to API.TEST.
Successfully added 'ServiceStack.Interfaces 4.0.50' to API.TEST.
Adding 'ServiceStack.Text 4.0.50' to API.TEST.
Successfully added 'ServiceStack.Text 4.0.50' to API.TEST.
Adding 'ServiceStack.Client 4.0.50' to API.TEST.
Successfully added 'ServiceStack.Client 4.0.50' to API.TEST.
Uninstalling 'ServiceStack.Client 4.0.48'.
Successfully uninstalled 'ServiceStack.Client 4.0.48'.
Uninstalling 'ServiceStack.Text 4.0.48'.
Successfully uninstalled 'ServiceStack.Text 4.0.48'.
Uninstalling 'ServiceStack.Interfaces 4.0.48'.
Successfully uninstalled 'ServiceStack.Interfaces 4.0.48'.
PM> Install-Package ServiceStack.HttpClient
Attention to resolve dependency: 'ServiceStack.Interfaces (> 4.0.50)'
```

Rysunek 3 Instalowanie pakietu ServiceStack.HttpClient

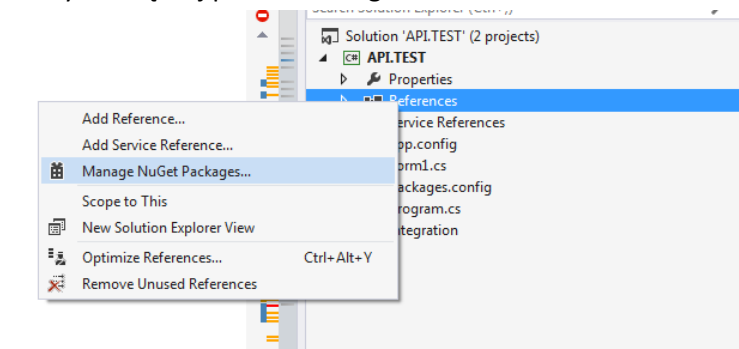
3. Po instalacji wymaganych pakietów w projekcie powinny pojawić się referencje do bibliotek wymienionych poniżej:
  - a. ServiceStack.Client
  - b. ServiceStack.HttpClient
  - c. ServiceStack.Interfaces
  - d. ServiceStack.Text



Rysunek 4 referencje ServiceStack client

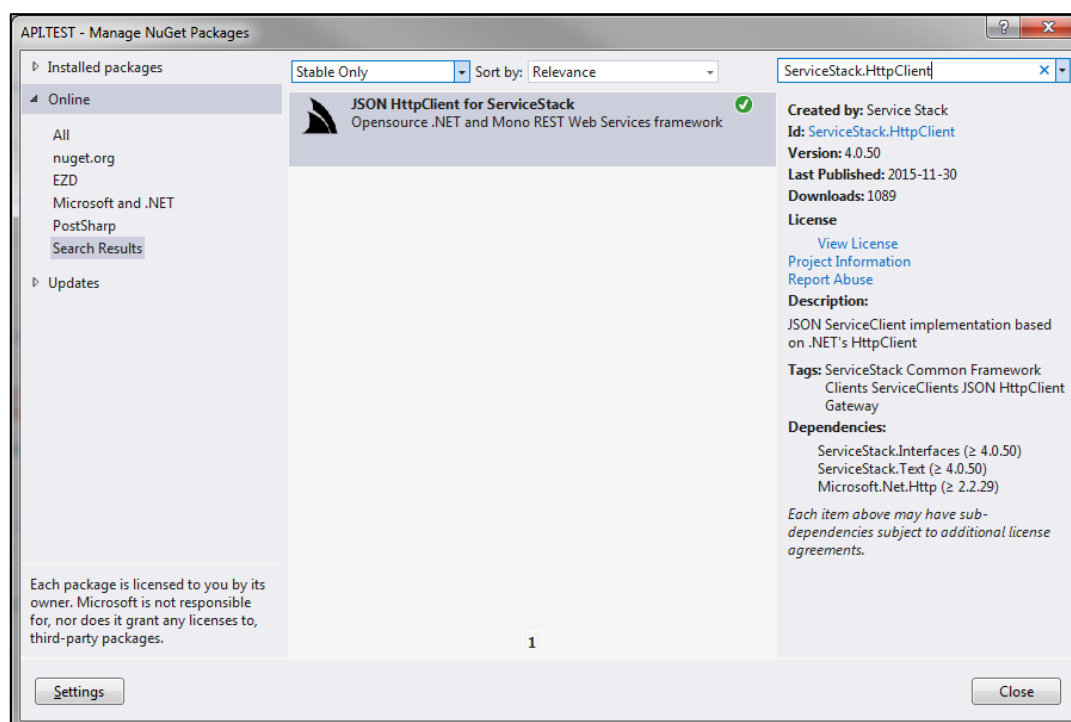
#### 4.1.1.2 Instalacja za pomocą menadżera pakietów NuGet:

1. Wskazujemy prawym klawiszem na kontenerze referencji w solucji i wybieramy „Manage NuGet Packages” czyli zarządzaj pakietami nuget.



Rysunek 5 Zarządzaj pakietami NuGet

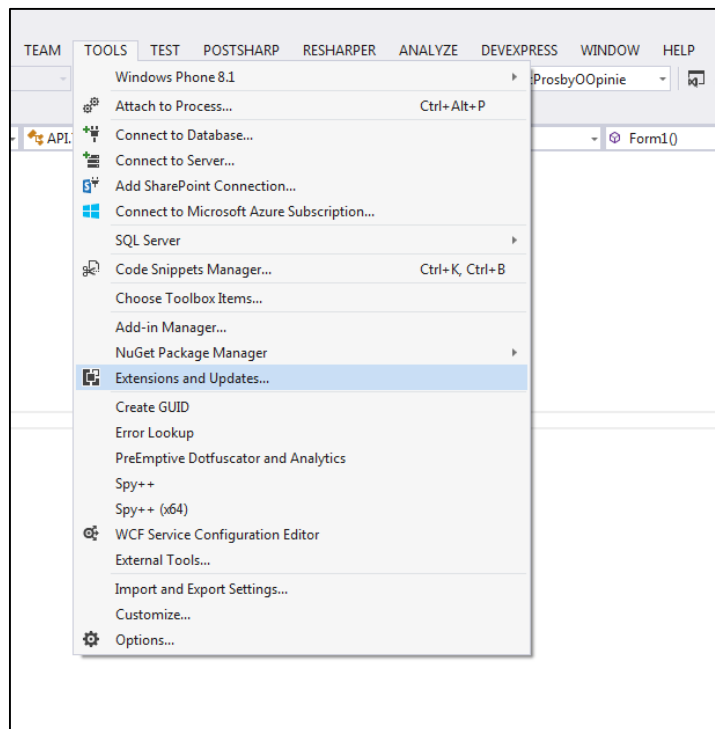
2. W konsoli Wyszukujemy pakiety „ServiceStack.Client” oraz „ServiceStack.HttpClient” i przyciskamy przycisk install w obydwu przypadkach referencje do pakietów powinny dołączyć się automatycznie.



Rysunek 6 Instalacja z GUI konsoli

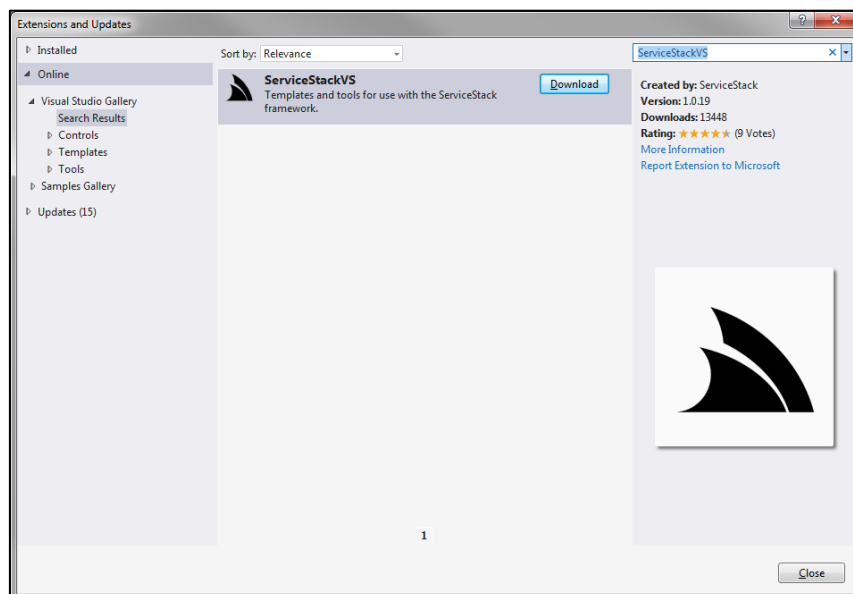
#### 4.1.1.3 Instalacja pakietu „ServiceStackVS” w Visual studio

1. W zakładce Tools znajduje się menager zarządzania dodatkami w Visual studio „Extensions and Updates” *Rysunek 7 Uruchomienie Extensions and Updates:*



**Rysunek 7 Uruchomienie Extensions and Updates**

2. Po uruchomieniu menagera wpisujemy w prawej górnej części okna polu wyszukiwania „ServiceStackVS” klikamy download.

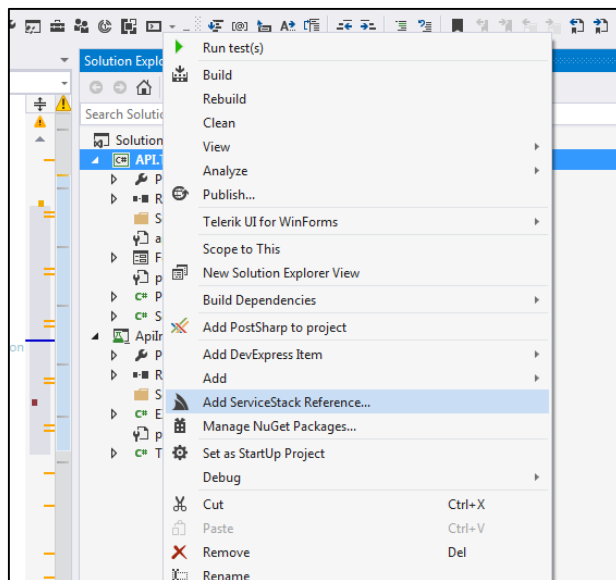


**Rysunek 8 Wyszukiwanie pakietu ServiceStackVS**

3. Po poprawnym zainstalowaniu pakietu oraz uruchomieniu ponownym VisualStudio powinien pakiet działać.

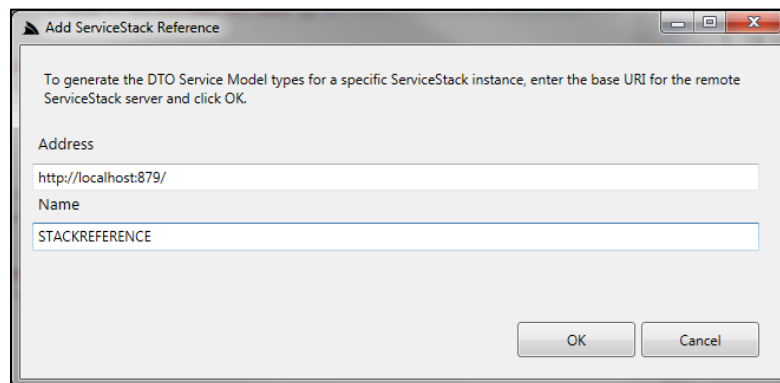
#### 4.1.1.4 Przygotowanie referencji DTO web serwisu do solucji z ServiceStack

1. Prawym klawiszem na solucji na liście z kontekstowego menu wskazujemy Add ServiceStack Reference.



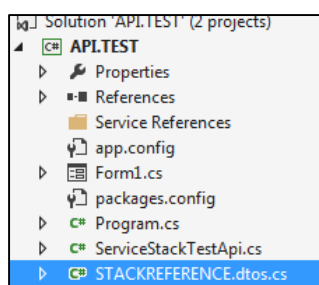
Rysunek 9 Dodawanie Referencji do solucji z Serwisu ServiceStack

2. Gdy pojawi się nowe okno w polu adres wpisujemy adres API ServiceStack oraz nazwę klasy w której będą przetrzymywane modele DTO.



Rysunek 10 Tworzenie referencji ServiceStack

3. Po synchronizacji api z solucją powinna pojawić się klasa o wybranej nazwie z przedrostkiem dto, posiada ona wszystkie DTO wykorzystywane do konsumowania serwisu.



Rysunek 11 Dto class

## 4.1.2 Przykłady komunikacji

Przykłady klientów z zadeklarowanym typem Endpointa (punktu końcowego) zmienna url określa adres serwisu API :

```
string url = @"http://<<adres hosta>>:<<port jeśli inny niż 80>>/" ;
/// domyślny endpoint Json
var client_json = new JsonServiceClient(url);
/// domyślny endpoint jsv
var client_jsv = new JsvServiceClient(url);
/// domyślny endpoint XML
var client_xml = new XmlServiceClient(url);
/// domyślny endpoint Soap 11
var clienat_soap11 = new Soap11ServiceClient(url);
/// domyślny endpoint Soap 12
var clienat_soap12 = new Soap12ServiceClient(url);
```

Rekomendowane jest korzystanie z klienta z endpointem typu JSON. Przykładowe pobranie z serwisu obiektu o odpowiednim typie przy wykorzystaniu różnych endpointów:

```
var koszulka_json = client_json.Post(new PobierzKoszulkePoIdRequest()
{ Id = 94114 });
var koszulka_jsv = client_jsv.Post(new PobierzKoszulkePoIdRequest()
{ Id = 94114 });
var response = clienat_soap11.Post<PobierzKoszulkePoIdResponse>("
/Koszulka/PoId/94114");
```

Przykłady metod zwracające w postaci zadeklarowanego typu oraz ich konwersja na obiekt DTO

```
/// Response w postaci JSON
string Response_raw_json = client_json.Post<string>(new
PobierzKoszulkePoIdRequest()
{ Id = 94114 });
/// konwersja do obiektu dto
var dto = Response_raw_json.FromJson<PobierzKoszulkePoIdResponse>();
```

Przykład wykonania aktualizacji obiektu koszulki:

```
var pr = client.Post(new AktualizujKoszulkeRequest()
{
    koszulka = new PismoDto() { ID = 51553, Nazwa = "Nowa nazwa koszulki"
}
}
) as AktualizujKoszulkeResponse;
```

Przykład autentykacji do serwisu API Elektronicznego zarządzania dokumentacją:

- Tworzenie tokena autentykacji z serwisem. Token składa się z 3 parametrów :
  - Guida wygenerowanego przez aplikację.
  - Tokena aplikacji wygenerowanego przez ezd w formacie SHA256.
  - Daty sformatowanej do stringa rok miesiąc dzień godzina „yyyyMMddhh”.



Przykład wykorzystania poniżej:

```
Guid parametrAutentykacji = Guid.NewGuid();/// Guid parametr do generowania tokena
string tokenAplikacji = "<<64 ZNAKOWY TOKEN APLIKACJI WYGENEROWANY W ezd>>"; /// Token
ustalany w ezd
string tokenAutentykacji = String.Format("{0}{1}{2}", parametrAutentykacji.ToString(),
tokenAplikacji, DateTime.Now.ToString("yyyyMMddhh")); /// Kreowanie tokena
jednorazowej autentykacji
SHA256 crypter = SHA256Managed.Create();// Inicjalizacja SHA256 obiektu
haszowania.
byte[] createToken = crypter.ComputeHash(Encoding.ASCII.GetBytes(tokenAutentykacji),
0, Encoding.ASCII.GetByteCount(tokenAutentykacji)); // Generowanie hasza na
podstawie wygenerowanych parametrów

string stringToken = string.Empty;
foreach (byte theByte in createToken)
{
    stringToken += theByte.ToString("x2");
} // generowanie stringa tokena jednorazowej komunikacji
```

- Dodawanie Nagłówek autentykacji każda opreacj musi mieć wygenerowany własne unikalne parametry. Przykład poniżej:

```
client.AddHeader("authParam", parametrAutentykacji.ToString());
client.AddHeader("authToken", stringToken);
```

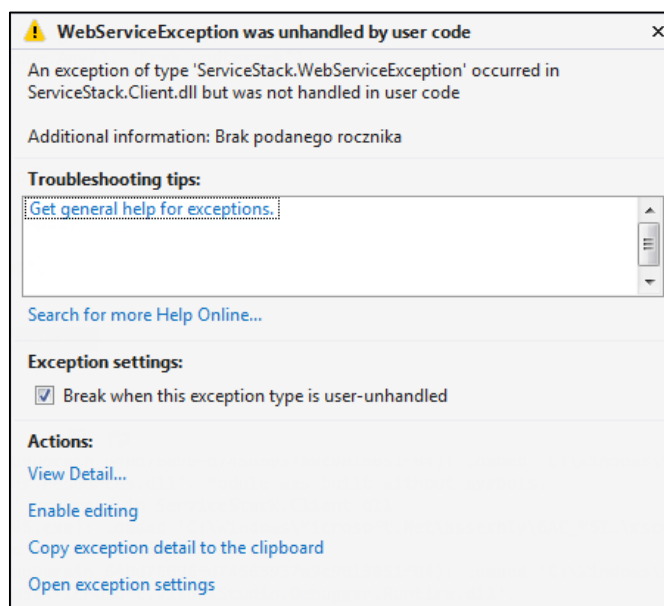
## 5 Opis interfejsów oraz ich wykorzystanie

### 5.1 Pobieranie całego drzewa Rwa z konkretnego rocznika

Metoda pobierze cały zestaw RWA o podanym roczniku:

```
var res = client.Post(new PobierzRwaPoRocznikuRequest()
{
    Rocznik = 2015
}) as PobierzRwaPoRocznikuResponse;
```

W przypadku nie podania prawidłowego rocznika wystąpi następujący wyjątek :



#### 5.1.1 DTO request – PobierzRwaPoRocznikuRequest

```
[Route("/Rwa/PobierzRwaPoRoczniku")]
public class PobierzRwaPoRocznikuRequest : ResponseBaseDto,
IReturn<PobierzRwaPoRocznikuResponse>
{
    public virtual int Rocznik { get; set; }
}
```

#### 5.1.2 DTO response – PobierzRwaPoRocznikuResponse

Obiekt przekazany wewnątrz `PobierzRwaPoRocznikuResponse` opisany jest w [6.5 TeczkaRwaDto](#)

```
public class PobierzRwaPoRocznikuResponse: ResponseBaseDto
{
    public virtual TeczkaRwaDto Teczki { get; set; }
}
```

## 5.2 Pobieranie wszystkich jednostek w strukturze

Ta metoda nie wymaga przekazywania żadnych parametrów przykład uruchomienia:

```
var res = client.Post(new PobierzWszytkieJednostkiRequest()) as
PobierzWszytkieJednostkiResponse;
```

## 5.3 DTO request – PobierzWszytkieJednostkiRequest

```
[Route("/Jednostka/PobierzWszytkie")]
public class PobierzWszytkieJednostkiRequest : RequestBaseDto,
IReturn<PobierzWszytkieJednostkiResponse>
{
}
```

## 5.4 DTO response – PobierzWszytkieJednostkiResponse

Obiekt zwraca listę obiektów JednostkaDto opisanych [6.4 JednostkaDto](#).

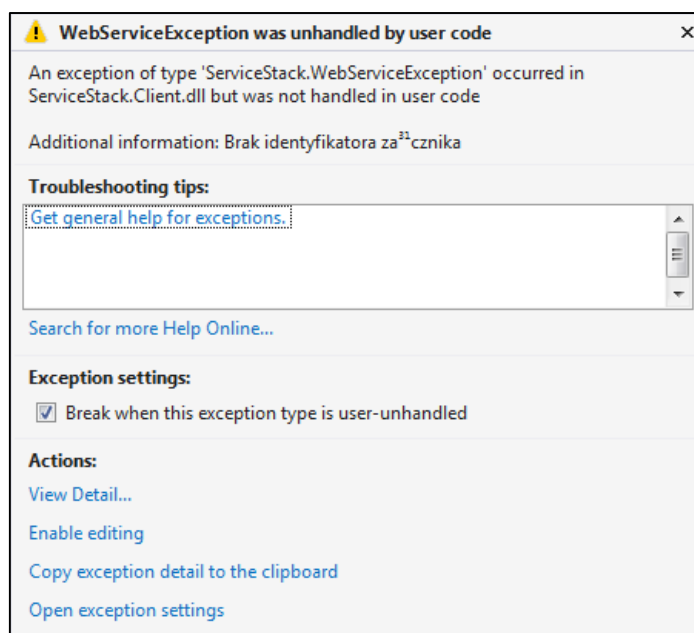
```
public class PobierzWszytkieJednostkiResponse : ResponseBaseDto
{
    public virtual List<JednostkaDto> Jednostki { get; set; }
}
```

## 5.5 Pobieranie załączników z storage

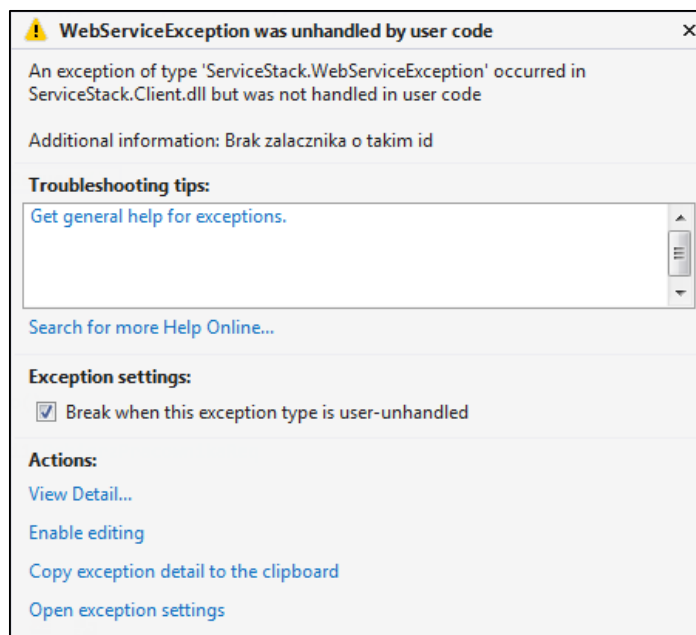
Aby pobrać załącznik ze storage wymagane jest podanie w obiekcie request identyfikatora załącznika:

```
var pr = client.Post(new PobierzZalacznikRequest()
{
    IdZalacznia =84226
}) as PobierzZalacznikResponse;
```

Przypadku nie podania identyfikatora załącznika wystąpi następujący wyjątek:



W przypadku gdy załącznik nie istnieje nastąpi wyjątek:



## 5.5.1 DTO request – PobierzZalacznikRequest

Reprezentacja obiektu dto przekazywanego postem aby wywołać metodę pobierania załącznika.

```
[Route("/Zalacznik/PobierzZalacznik")]
public class PobierzZalacznikRequest : RequestBaseDto,
IReturn<PobierzZalacznikResponse>
{
    public virtual int IdZalacznia { get; set; }
    public virtual string NazwaZalacznika { get; set; }
}
```

## 5.5.2 DTO response – PobierzZalacznikResponse

```
public class PobierzZalacznikResponse : ResponseBaseDto
{
    /// <summary>
    /// Informacje o złączniku
    /// </summary>
    public virtual ZalacznikDto zalacznikDto { get; set; }
    /// <summary>
    /// Załącznik w postaci binarnej
    /// </summary>
    public virtual byte[] zalacznik { get; set; }
    /// <summary>
    /// Nazwa załącznika z rozszerzeniem
    /// </summary>
    public virtual string Nazwa { get; set; }
}
```

## 5.6 Przenoszenie pracownika w strukturze

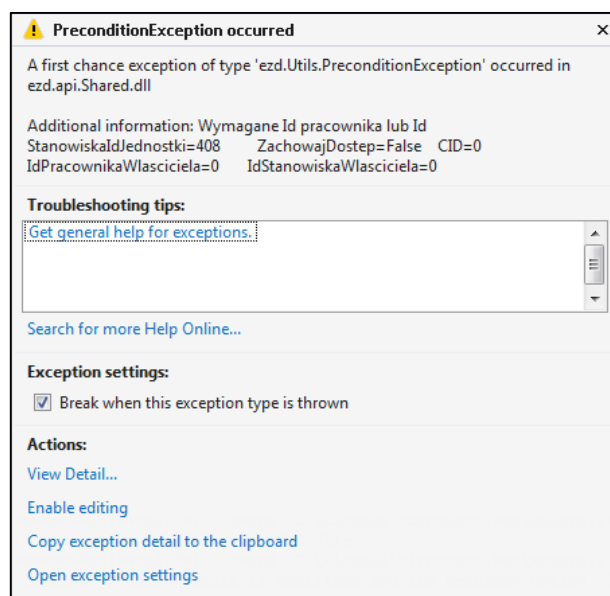
Aby przenieść pracownika w strukturze wymagane jest podanie identyfikatora jednostki oraz identyfikatora pracownika lub stanowiska.

Poniżej przykład przeniesienia pracownika w strukturze:

```
var res = client.Post(new PrzeniesPracownikaRequest()
{
    IdJednostki = 408,
    IdPracownikaWlasciciela = 646

}) as PrzeniesPracownikaResponse;
```

W przypadku nie podania prawidłowego identyfikatora jednostki nastąpi Exception poniżej:



### 5.6.1 DTO request – PrzeniesPracownikaRequest

Reprezentacja obiektu dto przekazywanego postem aby wywołać metodę przenoszenie pracownika.

```
[Route("/Pracownik/PrzeniesPracownika")]
public class PrzeniesPracownikaRequest : RequestBaseDto ,
IReturn<PrzeniesPracownikaResponse>
{
    public int IdJednostki { get; set; }
    public bool ZachowajDostep { get; set; }
}
```

### 5.6.2 DTO response – PrzeniesPracownikaResponse

Zwracany jest obiekt z identyfikatorem nowej jednostki.

```
public class PrzeniesPracownikaResponse: ResponseBaseDto
{
}
```



```
    public int IdJednostki { get; set; }  
}
```

## 5.7 Dodaj Pracownika

Aby dodać pracownika w systemie EZD wymaga się minimum poniższych parametrów w obiekcie `DodajPracownikaRequest`:

- Imie
- Nazwisko
- Stanowisko
- Identyfikator jednostki `IdJednostki`

Jeśli nie podamy loginu to pracownika pracownik się utworzy ale nie będzie można się na niego zalogować.

Przykład tworzenia pracownika w systemie :

```
var res = client.Post(new DodajPracownikaRequest()  
{  
    Imie = "Imie",  
    Nazwisko = "Nazwisko",  
    Login = "inazwisko",  
    Stanowisko = "stanowisko",  
    Inicjaly = "IM",  
    IdJednostki = 1,  
    ZmienHaslo = false,  
    Email = "email@prawidlowy.pl",  
    SortOrder = 0,  
  
}) as DodajPracownikaResponse;
```

### 5.7.1 DTO request – `DodajPracownikaRequest`

Reprezentacja obiektu dto przekazywanego postem aby wywołać metodę Dodawania pracownika.

```
[Route("/Pracownik/DodajPracownika")]  
public class DodajPracownikaRequest : RequestBaseDto,  
IReturn<DodajPracownikaResponse>  
{  
    /// <summary>  
    /// Imie pracownika  
    /// </summary>  
    public string Imie { get; set; }  
    /// <summary>  
    /// Nazwisko pacownika  
    /// </summary>  
    public string Nazwisko { get; set; }  
    /// inicjały pracownika  
    public string Inicjaly { get; set; }  
    /// <summary>  
    /// stanowisko pracownika  
    /// </summary>  
    public string Stanowisko { get; set; }  
    /// <summary>  
    /// Identyfikator jednostki  
    /// </summary>
```



```
public int IdJednostki { get; set; }
/// <summary>
/// Identyfikator wydziału pracownika
/// </summary>
public int? IdWydziału { get; set; }
/// <summary>
/// Pzy następnym logowaniu wymuszenie zmiany hasła
/// </summary>
public bool ZmienHaslo { get; set; }
/// <summary>
/// Urzytkownik ukryty w drzewie
/// </summary>
public bool Ukryty { get; set; }
/// <summary>
/// Login do logowania ręcznego
/// </summary>
public string Login { get; set; }
/// <summary>
/// Hasło do logowania ręcznego
/// </summary>
public string Haslo { get; set; }
/// <summary>
/// login activedirectory
/// </summary>
public string ActiveDirectory { get; set; }
/// <summary>
/// email
/// </summary>
public string Email { get; set; }
/// <summary>
/// Kolejność sortowania w drzewie
/// </summary>
public int? SortOrder { get; set; }
/// <summary>
/// Data ważności konta jeśli null to bezterminowa
/// </summary>
public DateTime? WazneDo { get; set; }
/// <summary>
/// Dodatkowe atrybuty określone przez jednostkę
/// </summary>
public string Atrybut1 { get; set; }
public string Atrybut2 { get; set; }
public string Atrybut3 { get; set; }
public string Atrybut4 { get; set; }
public string Atrybut5 { get; set; }
public string Atrybut6 { get; set; }
}
```

## 5.7.2 DTO response – DodajPracownikaResponse

```
public class DodajPracownikaResponse : ResponseBaseDto
{
    public int IdPracownika { get; set; } // identyfikator pracownika utworzonego
    pracownika

    public int IdStanowiska { get; set; } // identyfikator stanowiska utworzonego
    pracownika
}
```

## 5.8 Pobierz pracowników należących do jednostki niezależnie od typu jednostki.

Aby pobrać pracowników z danej jednostki należy podać id jednostki przykład :

```
var res = client.Post(new PobierzPracownikowNalezacychDoJednostkiRequest() { Id = 1 })
as PobierzPracownikowNalezacychDoJednoskiResponse;
```

### 5.8.1 DTO request – PobierzPracownikowNalezacychDoJednostkiRequest

Reprezentacja obiektu dto przekazywanego postem aby wywołać metodę pobierania pracowników należących do jednostki.

```
[Route("/Pracownik/PobierzPracownikowNalezacychDowydzialuLubOddzialu")]
public class PobierzPracownikowNalezacychDoJednostkiRequest : RequestBaseDto,
IReturn<PobierzPracownikowNalezacychDoJednoskiResponse>
{
    public virtual long Id { get; set; }
}
```

### 5.8.2 DTO response – PobierzPracownikowNalezacychDoJednostkiResponse

W obiekcie zostaje zwrócona lista obiektów typu PracownikDto opisana [6.3 PracownikDto](#)

```
public class PobierzPracownikowNalezacychDoJednoskiResponse : ResponseBaseDto
{
    public virtual List<PracownikDto> Pracownicy { get; set; }
}
```

## 5.9 Pobierz Pracownika

Pobieranie pracownika daje różne możliwości i warianty pobierania/wyszukiwania osób w strukturze . Poniżej przedstawiono przykładowe kombinacje wyszukiwania pracownika w strukturze organizacyjnej:

1. Wyszukanie pracownika w przypadku kiedy znamy tylko pełnioną rolę organizacyjną oraz w jakim wydziale jest (po roli organizacyjnej oraz identyfikatorze wydziału):

```
var res = serv.GetPracownik(new PobierzPracownikaReq()
{
    RoleOrganizacyjneKlucz = "DYREKTOR",
    IdJednostkiZrodlo = 20
});
```

2. Wyszukanie przełożonego pracownika :

```
var res = client.Post(new ezd.Data.Flow.Api1.PobierzPracownikaReq()
{
    RoleOrganizacyjneKlucz = "PRZELOZONY",
    IdPracownikaZrodlo = 881
});
```

3. Wyszukiwanie za pomocą samej roli organizacyjnej :

```
var res = client.Post(new ezd.Data.Flow.Api1.PobierzPracownikaReq()
{
    RoleOrganizacyjneKlucz = "DYREKTORGESERALNY",
});
```

4. Wyszukaj pracownika po id

```
var res = client.Post(new ezd.Data.Flow.Api1.PobierzPracownikaReq()
```





```
{  
    IdPracownikaZrodlo = 448,  
});
```

5. Wyszukaj pracownika po id stanowiska

```
var res = client.Post(new ezd.Data.Flow.Api1.PobierzPracownikaReq()  
{  
    IdStanowiskaZrodlo = 367,  
});
```

## 5.9.1 DTO request – PobierzPracownikaReq

```
[Route("/Pracownik/PobierzPracownika")]  
[Route("/api1/PobierzPracownika")]  
[DataContract(Namespace = Constants.EZD_NAMESPACE_DATA, Name =  
"PobierzPracownikaReq")]  
public class PobierzPracownikaReq  
{  
  
    /// <summary>  
    /// Role organizacyjne które są wymagane przy szukany pracowniku  
    /// Role rozdzielone średnikiem, np: Dyrektor Wydziału;Kierownik Oddziału  
    /// </summary>  
    [DataMember]  
    public string RoleOrganizacyjne { get; set; }  
  
    /// <summary>  
    /// Role organizacyjne które są wymagane przy szukany pracowniku  
    /// </summary>  
    [DataMember]  
    public string[] RoleOrganizacyjneArray { get; set; }  
  
    /// <summary>  
    /// Role organizacyjne które są wymagane przy szukany pracowniku  
    /// Role rozdzielone średnikiem, np: Dyrektor Wydziału;Kierownik Oddziału  
    /// </summary>  
    [DataMember]  
    public string RoleOrganizacyjneKlucz { get; set; }  
  
    /// <summary>  
    /// Role organizacyjne które są wymagane przy szukany pracowniku  
    /// </summary>  
    [DataMember]  
    public string[] RoleOrganizacyjneKluczArray { get; set; }  
  
    /// <summary>  
    /// IdStanowiska osoby w kontekście którego szukany jest pracownik  
    /// </summary>  
    [DataMember]  
    public int IdStanowiskaZrodlo { get; set; }  
  
    /// <summary>  
    /// IdPracownika osoby w kontekście której szukany jest pracownik  
    /// </summary>  
    [DataMember]  
    public int IdPracownikaZrodlo { get; set; }  
  
    /// <summary>  
    /// Typ jednostki w której będzie szukany pracownik docelowy
```

```
    /// Typ jednostki szukany (w górę struktury) jest na podstawie pracownika
    ŹRÓDŁO
    /// ,następnie docelowy pracownik szukany jest (w dół) w strukturach
    wuyszukanej jednostki
    /// </summary>
    [DataMember]
    public string TypJednostkiPracownikaZrodlo { get; set; }

    /// <summary>
    /// Symbol jednostki w której szukany jest pracownik
    /// </summary>
    [DataMember]
    public string SymbolJednostkiZrodlo { get; set; }

    /// <summary>
    /// ID jednostki w której szukany jest pracownik
    /// </summary>
    [DataMember]
    public int IdJednostkiZrodlo { get; set; }

    /// <summary>
    /// Nazwa stanowiska którego szukamy
    /// </summary>
    [DataMember]
    public string NazwaStanowiska { get; set; }

    /// <summary>
    /// Nazwa stanowiska którego szukamy
    /// </summary>
    [DataMember]
    public string NazwaJednostki { get; set; }
}
```



## 5.9.2 DTO response – PobierzPracownikaRes

```
public class PobierzPracownikaRes
{
    /// <summary>
    /// Główny identyfikator pracownika
    /// </summary>
    [DataMember]
    public int IdPracownika { get; set; }
    /// <summary>
    /// Originalny identyfikator przed modyfikacjami
    /// </summary>
    [DataMember]
    public int IdPracownikaOriginal { get; set; }

    /// <summary>
    /// Główny identyfikator stanowiska
    /// </summary>
    [DataMember]
    public int IdStanowiska { get; set; }
    /// <summary>
    /// Originalny identyfikator przed modyfikacjami
    /// </summary>
    [DataMember]
    public int IdStanowiskaOriginal { get; set; }

    /// <summary>
    /// Identyfikator jednostki do której pracownik przynależy
    /// </summary>
    [DataMember]
    public int IdJednostki { get; set; }
    /// <summary>
    /// Originalny Identyfikator jednostki do której pracownik przynależy
    /// </summary>
    [DataMember]
    public int IdJednostkiOriginal { get; set; }
    /// <summary>
    /// Nazwa wyświetlana jednostki
    /// </summary>
    [DataMember]
    public string NazwaJednostki { get; set; }
    /// <summary>
    /// Symbol jednostki
    /// </summary>
    [DataMember]
    public string SymbolJednostki { get; set; }

    /// <summary>
    /// Identyfikator wydziału jeśli w jakimś przebywa
    /// </summary>
    [DataMember]
    public int IdWydzial { get; set; }
    /// <summary>
    /// Originalny Identyfikator wydziału jeśli w jakimś przebywa
    /// </summary>
    [DataMember]
```



```
public int IdWydzialOriginal { get; set; }

/// <summary>
/// Nazw wyświetlana wydziału
/// </summary>
[DataMember]
public string NazwaWydzialu { get; set; }

/// <summary>
/// Symbol wydziału
/// </summary>
[DataMember]
public string SymbolWydzialu { get; set; }

/// <summary>
/// Imię i nazwisko pracownika
/// </summary>
[DataMember]
public string ImieNazwisko { get; set; }

/// <summary>
/// Nazwa stanowiska
/// </summary>
[DataMember]
public string Stanowisko { get; set; }
}
```

## 5.10 Pobierz wszystkie identyfikatory dokumentów znajdujące się w koszulce

Aby pobrać identyfikatory wymagane jest podanie id koszulki przykład pobrania poniżej:

```
var res = client.Post(new
    PobierzIdentyfikatoryDokumentowKoszulkiRequest
    {
        IdKoszulki = 95607,
    }) as PobierzIdentyfikatoryDokumentowKoszulkiResponse;
```

### 5.10.1 DTO request - PobierzIdentyfikatoryDokumentowKoszulkiRequest

```
[Route("/Dokument/PobierzIdentyfikatoryDokumentowKoszulki")]
public class PobierzIdentyfikatoryDokumentowKoszulkiRequest : RequestBaseDto,
    IReturn<PobierzIdentyfikatoryDokumentowKoszulkiResponse>
{
    public virtual int IdKoszulki { get; set; }
}
```

### 5.10.2 DTO response - PobierzIdentyfikatoryDokumentowKoszulkiResponse

```
public class PobierzIdentyfikatoryDokumentowKoszulkiResponse : ResponseBaseDto
{
    public virtual List<long> dokumenty { get; set; }
}
```

## 5.11 Dodawanie załączników

Aby dodać załącznik do systemu ezd wymagane jest :

- Dane czyli załącznik w bajtach
- Nazwa z rozszerzeniem załącznika
- Identyfikator pracownika lub stanowiska

Przykład przesłania załącznika do systemu EZD:

```
string nazwa_pliku = @"1.jpg";
FileInfo informacje_o_pliku = new FileInfo(nazwa_pliku);
var pr = client.Post(new DodajZalacznikRequest()
{
    Dane = File.ReadAllBytes(nazwa_pliku),
    Nazwa = informacje_o_pliku.Name,
    IdPracownikaWlasciciela=448,
}) as DodajZalacznikResponse;
```

### 5.11.1 DTO request – DodajZalacznikRequest

```
[Route("/Zalacznik/DodajZalacznik")]
```



```
public class DodajZalacznikRequest : RequestBaseDto,
IReturn<DodajZalacznikResponse>
{
    /// <summary>
    /// Binarne dane załącznika
    /// </summary>
    public virtual byte[] Dane { get; set; }
    /// <summary>
    /// Nazwa pliku z rozszerzeniem
    /// </summary>
    public virtual string Nazwa { get; set; }
}
```

## 5.11.2 DTO response – DodajZalacznikResponse

```
public class DodajZalacznikResponse : ResponseBaseDto
{
    /// <summary>
    /// Identyfikator kontentu
    /// </summary>
    public int ContentId { get; set; }
    /// <summary>
    /// Nazwa załącznika
    /// </summary>
    public string Nazwa { get; set; }
}
```

## 5.12 Aktualizacja informacji o dokumencie

Aby wykonać aktualizację dokumentu wymagane jest przekazanie nie jest to załącznik :

- WskazanieDokumentuDto w którym wypełnione powinno być Identyfikator (int) lub IdentyfikatorDokumentu (string) bez jednego lub drugie parametru nie można zlokalizować dokumentu.
- Wymagane jest podanie IdPracownikaWlasciciela oraz IdStanowiskaWlasciciela

Przykładowa aktualizacja nazwy obiektu dokument :

```
var res = serv.AktualizujDokument(new AktualizujDokumentReq()
{
    Dokument = new DokumentSystemowyTypeDto()
    {
        Identyfikator = new WskazanieDokumentuDto()
        {
            Identyfikator = 642103,
        },
        Nazwa = "Nazwa Testowa.jpg"
    },
    IdPracownikaWlasciciela = 448,
    IdStanowiskaWlasciciela = 448
}) as AktualizujDokumentRes;
```



## 5.12.1 DTO request – AktualizujDokumentReq

Data Transfer object do wywołania aktualizacji dokumentu:

```
public class AktualizujDokumentReq : RequestBaseDto, IReturn<AktualizujDokumentRes>
{
    /// <summary>
    /// Obiekt dokumentu który dziedziczy po public class JednostkaDto :
    ResponseBaseDto
    {
        /// <summary>
        /// Identyfikator jednostki
        /// </summary>
        public virtual int ID { get; set; }
        /// <summary>
        /// Nazwa jednostki
        /// </summary>
        public virtual string Nazwa { get; set; }
        /// <summary>
        /// Opis jednostki
        /// </summary>
        public virtual string Opis { get; set; }
        /// <summary>
        /// Symbol jednostki
        /// </summary>
        public virtual string Symbol { get; set; }
        /// <summary>
        /// Czy jednostka jest aktywna
        /// </summary>
        public virtual bool Aktywny { get; set; }
        /// <summary>
        /// Kolejność sortowania
        /// </summary>
        public virtual int SortOrder { get; set; }
        /// <summary>
        /// Data dodania jednostki do organizacji
        /// </summary>
        public virtual DateTime DataDodania { get; set; }
        /// <summary>
        /// Id nadrzędne jednostki jeśli jest Rootem jednostki zawsze będzie null
        /// jeśli nie to określa jaka jednostka jest nad nią
        /// </summary>
        public virtual int? IdNadrzedne { get; set; }
        public virtual int? IdPoprzedniego { get; set; }
        /// <summary>
        /// Typ jednostki Jednostka / wydział / oddział
        /// </summary>
        public virtual int IdTypJednostki { get; set; }
        /// <summary>
        /// Id wydziału
        /// </summary>
        public virtual int? IdWydzialu { get; set; }
        /// <summary>
        /// Id wydziału orginał
        /// </summary>
        public virtual int? IdWydzialuOrginal { get; set; }
        /// <summary>
        /// Dodatkowe atrybuty jednostki
        /// </summary>
        public virtual string Atrybut1 { get; set; }
```



```
    public virtual string Atrybut2 { get; set; }  
    public virtual string Atrybut3 { get; set; }  
}
```

## 5.13 TeczkaRwaDto

```
public class TeczkaRwaDto  
{  
    /// <summary>  
    /// Identyfikator teczki  
    /// </summary>  
    public virtual int ID { get; set; }  
    /// <summary>  
    /// Teczki podrzędne  
    /// </summary>  
    public virtual List<TeczkaRwaDto> TeczkiPodrzedne { get; set; }  
    /// <summary>  
    /// Rocznik teczki  
    /// </summary>  
    public virtual int Rok { get; set; }  
    /// <summary>  
    /// Symbol teczki  
    /// </summary>  
    public virtual string Symbol { get; set; }  
    /// <summary>  
    /// Nazwa wyświetlana teczki  
    /// </summary>  
    public virtual string Nazwa { get; set; }  
    /// <summary>  
    /// Kategoria archiwalna  
    /// </summary>  
    public virtual string KategoriaArchiwalna { get; set; }  
    /// <summary>  
    /// Typ prowadzenia Elektroniczna = 0, Tradycyjna = 1  
    /// </summary>  
    public virtual TypProwadzenia TypProwadzenia { get; set; }  
    public virtual int IloscWyjatkow { get; set; }  
    public virtual bool UdostepnienieWyjatek { get; set; }  
    public virtual int NumerLP { get; set; }  
    public virtual int? LP { get; set; }  
    /// <summary>  
    /// Czy teczka jest aktywna  
    /// </summary>  
    public virtual bool Nieaktywny { get; set; }  
    /// <summary>  
    /// Ilość dni na załatwienie sprawy w danej kategorii  
    /// </summary>  
    public virtual int? TerminDni { get; set; }  
}  
  
DokumentTypeDto  
    /// </summary>  
    [DataMember]  
    public DokumentSystemowyTypeDto Dokument { get; set; }  
}
```



### 5.13.1 DTO response – AktualizujDokumentRes

```
public class AktualizujDokumentRes : ResponseBaseDto
{
    /// <summary>
    /// Identyfikator dokumentu
    /// </summary>
    [DataMember]
    public long IdDokumentu { get; set; }
}
```

## 5.14 Pobieranie identyfikatorów koszulek których właścicielem jest określony Pracownik.

Wymaganymi parametrami aby pobrać identyfikatory koszulek pracownika są :

- Id właściciela lub id stanowiska właściciela

Przykładowe wywołanie pobrania identyfikatorów koszulek pracownika po id pracownika:

```
var res = serv.PobierzIdentyfikatoryKoszulekPracownika(new
PobierzIdentyfikatoryKoszulekRequest()
{
    IdPracownikaWlasciciela = 448
}) as PobierzIdentyfikatoryKoszulekResponse;
```

### 5.14.1 DTO request – PobierzIdentyfikatoryKoszulekRequest

```
[Route("/Koszulka/PobierzIdentyfikatoryKoszulek")]
public class PobierzIdentyfikatoryKoszulekRequest : RequestBaseDto,
IReturn<PobierzIdentyfikatoryKoszulekResponse>
{
    [DataMember]
    public int IdPracownikaWlasciciela { get; set; }
    [DataMember]
    public int IdStanowiskaWlasciciela { get; set; }
}
```

### 5.14.2 DTO response – PobierzIdentyfikatoryKoszulekResponse

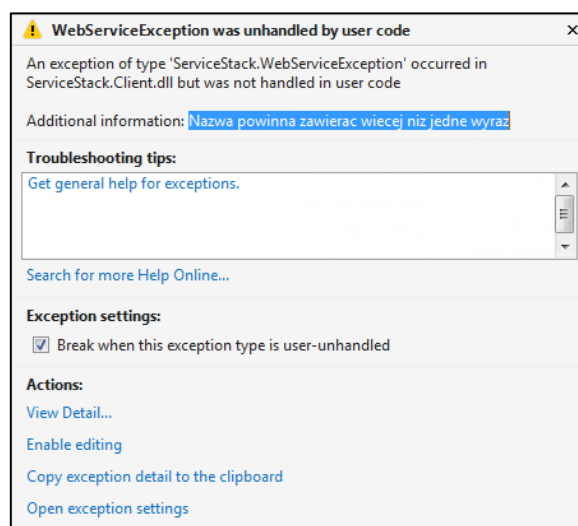
```
public class PobierzIdentyfikatoryKoszulekResponse: ResponseBaseDto
{
    /// <summary>
    /// Lista identyfikatorów koszulek
    /// </summary>
    public virtual List<int> Koszulki { get; set; }
}
```

## 5.15 Aktualizacja danych koszulki

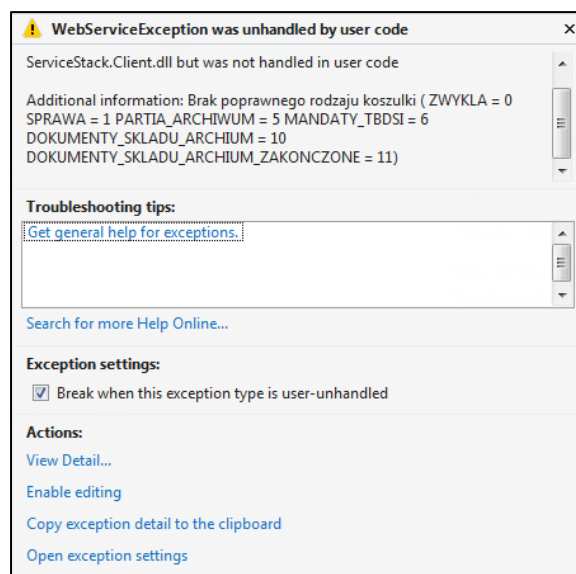
Aktualizuj koszulkę akceptuje tylko operacje typu Post. Wymagania dotyczące zmiennych aktualizacji:



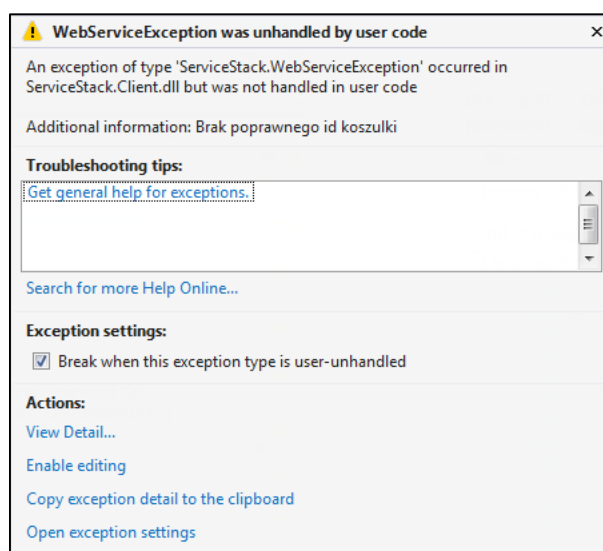
- PismoDto.ID – zmienna wymagana aby dokonać operacji aktualizacji w przypadku niewpisania **Rysunek 14 Wyjątek id koszulki PismoDto**
- PismoDto.Nazwa – (nie jest wymagana) powinna posiadać więcej niż jeden wyraz w przypadku kiedy nie spełni warunku nastąpi wyjątek **Rysunek 12 Wyjątek Nazwa PismoDto**.
- PismoDto.Rodzaj – (nie jest wymagana) Rodzaj powinien być generowany na podstawie enumeratora **RodzajKoszulki** jeśli warunek nie zostanie spełniony nastąpi wyjątek **Rysunek 13 Wyjątek Rodzaj PismoDto**.
- PismoDto.DataZakonczenia – (nie jest wymagana) nie można przypisywać wstecznej wartości jeśli warunek nie zostanie spełniony nastąpi wyjątek **Rysunek 15 Wsteczna wartość daty zakończenia koszulki**
- PismoDto.DataUtworzenia – zmienna nie jest możliwa do aktualizacji
- PismoDto.TerminPisma - (nie jest wymagana) termin pisma musi być większy od daty wpływu jeśli warunek nie zostanie spełniony nastąpi wyjątek **Rysunek 16 Wsteczna data terminu pisma**.



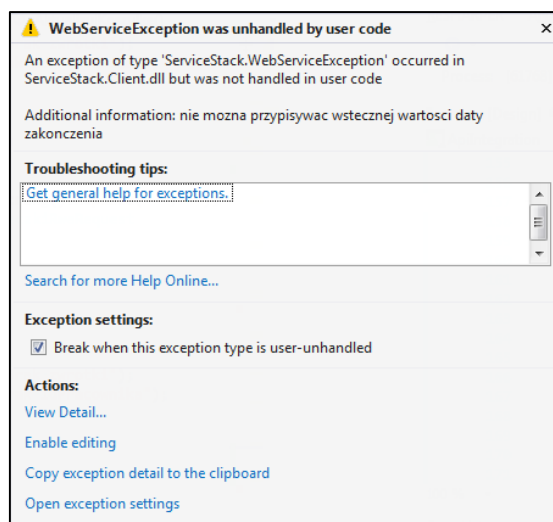
Rysunek 12 Wyjątek Nazwa PismoDto



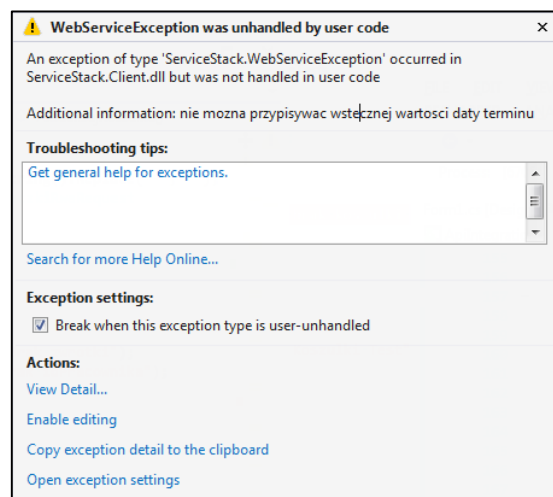
Rysunek 13 Wyjątek Rodzaj PismoDto



Rysunek 14 Wyjątek id koszulki PismoDto



Rysunek 15 Wsteczna wartość daty zakończenia koszulki



Rysunek 16 Wsteczna data terminu pisma

Przykładowa aktualizacja nazwy koszulki oraz rodzaju w języku c# wynikiem jest obiekt `AktualizujKoszulkeResponse`:

```
var pr = client.Post(new AktualizujKoszulkeRequest()
{
    koszulka = new PismoDto()
    {
        ID = 51553, Nazwa = "Nazwa Koszulki", Rodzaj = 0
    },
}) as AktualizujKoszulkeResponse;
```



## 5.15.1 DTO request – AktualizujKoszulkeRequest

Obiekt DTO żądania aktualizacji koszulki składa się z zmiennej koszulka która jest obiektem typu *PismoDto (Koszulka)*. Wszelkie zmiany koszulki wykonujemy na obiekcie koszulka typu PismoDto.

```
[Route("/Koszulka/AktualizujKoszulke")]
public class AktualizujKoszulkeRequest : IReturn<AktualizujKoszulkeResponse>
{
    /// <summary>
    /// Pismo/koszulka do aktualizacji
    /// </summary>
    public virtual PismoDto koszulka { get; set; }
}
```



## 5.15.2 DTO response – AktualizujKoszulkeResponse

Obiekt odpowiedzi żądania składa się z zmiennej koszulka która jest obiektem typu *PismoDto* (*Koszulka*).

```
public class AktualizujKoszulkeResponse
{
    /// <summary>
    /// Pismo/koszulka po aktualizacji
    /// </summary>
    public virtual PismoDto koszulka { get; set; }
}
```



## 5.16 PobierzWszytskichPracownikow

Pobranie wszystkich pracowników w instytucji przykład użycia:

```
var pr = client.Post(new PobierzWszytskichPracownikowRequest ()  
    {  
    }) as PobierzWszytskichPracownikowResponse;
```

Zostanie zwrócona lista obiektów dto „PracownikDto” reprezentujących pracowników instytucji.

### 5.16.1 DTO request – PobierzWszytskichPracownikowRequest

PobierzWszytskichPracownikowRequest obiekt nie posiada żadnych parametrów.

```
public class PobierzWszytskichPracownikowRequest :  
    IReturn<List<PobierzWszytskichPracownikowResponse>>  
    {  
    }
```

### 5.16.2 DTO response – PobierzWszytskichPracownikowResponse

Po wykonaniu zapytania zostanie zwrócona lista obiektów dto „PracownikDto”.

```
public class PobierzWszytskichPracownikowResponse : ResponseBaseDto  
    {  
        public virtual List<PracownikDto> Pracownicy { get; set; }  
    }
```

## 5.17 PrzekazKoszulke

### 5.17.1 DTO request – PrzekazKoszulkeReq

```
[Route("/Koszulka/PrzekazKoszulkeReq")]  
[Route("/Api1/PrzekazKoszulkeReq")]  
[DataContract(Namespace = Constants.EZD_NAMESPACE_DATA, Name =  
    "PrzekazKoszulkeReq")]  
public class PrzekazKoszulkeReq : IReturn<PrzekazKoszulkeRes>  
    {  
        /// <summary>  
        /// Identyfikator koszulki która będzie przekazywana  
        /// </summary>  
        [DataMember]  
        public int IdKoszulki { get; set; }  
  
        /// <summary>  
        /// Znak sprawy na podstawie którego będzie przekazywana  
        /// </summary>  
        [DataMember]  
        public string ZnakPisma { get; set; }  
  
        /// <summary>  
        /// Identyfikator stanowiska Właściciela  
        /// </summary>  
        [DataMember]  
        public int IdStanowiskaWlasciciela { get; set; }  
  
        /// <summary>
```

```
/// Identyfikator pracownika właściciela
/// </summary>
[DataMember]
public int IdPracownikaWlasciciela { get; set; }

/// <summary>
/// Identyfikator pracownika zrodlowego
/// </summary>
[DataMember]
public int IdPracownikaZrodlowego { get; set; }

/// <summary>
/// Identyfikator pracownika zrodlowego
/// </summary>
[DataMember]
public int IdStanowiskaZrodlowego { get; set; }

/// <summary>
/// Identyfikator pracownika docelowego
/// </summary>
[DataMember]
public int IdPracownikaDocelowego { get; set; }

/// <summary>
/// Identyfikator stanowiska docelowego
/// </summary>
[DataMember]
public int IdStanowiskaDocelowego { get; set; }
}
```



### 5.17.2 DTO response – PrzekazKoszulkeRes

Obiekt dto zwraca identyfikator etapu pisma.

```
public class PrzekazKoszulkeRes : ResponseBaseDto
{
    [DataMember]
    public int IdEtapPisma { get; set; }
}
```

### 5.18 RejestrujSprawe

Przykłady rejestrowania sprawy :

- Rejestrowanie sprawy po roczniku , symbolu teczki i id Koszulki

```
var res = client.Post (new RejestrujSpraweReq()
{
    Rok = 2015,
    IdPracownikaWlasciciela = 448,
    TeczkaSymbol = "000",
    IdKoszulki = koszulka.IdKoszulki,

    }) as RejestrujSpraweRes;
```

- Rejestrowanie sprawy po symbolu teczki idkoszulki DataRozpoczecia (rocznik będzie pobrany domyślnie)

```
var res = client.Post(new RejestrujSpraweReq()
{
    IdPracownikaWlasciciela = 448,
    TeczkaSymbol = "000",
    IdKoszulki = koszulka.IdKoszulki,
    Uwagi = "Uwaga",
    DataRozpoczecia = DateTime.Now.ToString()
    }) as RejestrujSpraweRes;
```

#### 5.18.1 DTO request – RejestrujSpraweReq

```
[Route("/RejestrSpraw/RejestrujSprawe")]
[Route("/Api3/RejestrujSpraweReq")]
[DataContract(Namespace = Constants.EZD_NAMESPACE_DATA + "api3", Name =
"RejestrujSpraweReq")]
public class RejestrujSpraweReq: RequestBaseDto, IReturn<RejestrujSpraweRes>
{
    [DataMember]
    public int NumerKolejnySprawy { get; set; }

    [DataMember]
    public string Uwagi { get; set; }
```

```
[DataMember]
public string ZnakSprawy { get; set; }

[DataMember]
public int IdKoszulki { get; set; }

[DataMember]
public int Rok { get; set; }

[DataMember]
public string TeczkaSymbol { get; set; }

[DataMember]
public int TeczkaId { get; set; }

[DataMember]
public string KategoriaArchiwalna { get; set; }

[DataMember]
public int IdProwadzacegoSprawe { get; set; }

[DataMember]
public int IdRejestrujacegoSprawe { get; set; }

[DataMember]
public string TypProwadzenia { get; set; }

[DataMember]
public int IdPracownikaWlasciciela { get; set; }

[DataMember]
public int IdStanowiskaWlasciciela { get; set; }

[DataMember]
public string DataRozpoczecia { get; set; }

[DataMember]
public int CID { get; set; }
}
```

### 5.18.2 DTO response – RejestrujSpraweRes

Obiekt zwracany po prawidłowej rejestracji sprawy w systemie:

```
public class RejestrujSpraweRes : ResponseBaseDto
{
    [DataMember]
    public int IdSprawy { get; set; }

    [DataMember]
    public DateTime DataRejestracjiSprawy { get; set; }

    [DataMember]
    public int IdTeczki { get; set; }

    [DataMember]
    public string SymbolTeczki { get; set; }

    [DataMember]
    public int IdTeczkiParent { get; set; }

    [DataMember]
    public string SymbolTeczkiParent { get; set; }

    [DataMember]
    public string KategoriaArchiwalna { get; set; }

    [DataMember]
    public string TypProwadzenia { get; set; }
}
```

### 5.19 UtworzKoszulke

Aby utworzyć koszulkę w systemie wymaga się minimum podanie:

- IdPracownikaWlasciciela lub IdStanowiskaWlasciciela
- Nazwa koszulki składająca się co najmniej z 2 Wyrazów

Przykład Tworzenia koszulki:

```
var res = client.Post(new UtworzKoszulkeReq()
{
    Nazwa = "Nowa koszulka",
    IdPracownikaWlasciciela = 448
});
```



## 5.19.1 DTO request – UtworzKoszulkeReq

```
public class UtworzKoszulkeReq : RequestBaseDto, IReturn<UtworzKoszulkeRes>
{
    /// <summary>
    ///     Nazwa koszulki
    /// </summary>

    [DataMember]
    public string Nazwa { get; set; }

    /// <summary>
    ///     Identyfikator Właściciela koszulki
    /// </summary>

    [DataMember]
    public int IdPracownikaWlasciciela { get; set; }

    /// <summary>
    ///     Identyfikator stanowiska Właściciela koszulki
    /// </summary>

    [DataMember]
    public int IdStanowiskaWlasciciela { get; set; }
}
```



## 5.19.2 DTO response – UtworzKoszulkeRes

```
public class UtworzKoszulkeRes: ResponseBaseDto
{
    /// <summary>
    ///     Identyfikator koszulki zarejestrowanej
    /// </summary>

    [DataMember]
    public int IdKoszulki { get; set; }
}
```

## 5.20 ZakonczKoszulke

Zakańczanie koszulki w systemie EZD wymagania:

- Identyfikator koszulki lub znak sprawy
- Id Stanowiska lub id Pracownika

Przykład zakańczania koszulki po jej identyfikatorze:

```
var res = client.Post(new ZakonczKoszulkeReq()  
    {  
        IdKoszulki = 2785,  
        IdPracownikaWlasciciela = 448  
    }) as ZakonczKoszulkeRes;;
```

Przykład zakańczania koszulki po jej znaku sprawy :

```
var res = client.Post(new ZakonczKoszulkeReq()  
    {  
        ZnakSprawy= "BI.III.000.1714.2015",  
        IdPracownikaWlasciciela = 448  
    }) as ZakonczKoszulkeRes;;
```

### 5.20.1 DTO request – ZakonczKoszulkeReq

```
public class ZakonczKoszulkeReq : RequestBaseDto, IReturn<ZakonczKoszulkeRes>  
{  
    /// <summary>  
    /// Identyfikator koszulki  
    /// </summary>  
    [DataMember]  
    public int IdKoszulki { get; set; }  
    /// <summary>  
    /// Id Stanowiska Właściciela Koszulki  
    /// </summary>  
    [DataMember]  
    public int IdStanowiskaWlasciciela { get; set; }  
    /// <summary>  
    /// Id Pracownika Właściciela Koszulki  
    /// </summary>  
    [DataMember]  
    public int IdPracownikaWlasciciela { get; set; }  
    /// <summary>  
    /// Znak sprawy  
    /// </summary>  
    [DataMember]  
    public string ZnakSprawy { get; set; }  
}
```

### 5.20.2 DTO response – ZakonczKoszulkeRes

```
[DataContract(Namespace = Constants.EZD_NAMESPACE_DATA, Name =
"ZakonczKoszulkeRes")]
public class ZakonczKoszulkeRes : ResponseBaseDto
{
    [DataMember]
    public int IdEtapPisma { get; set; }
}
```

## 5.21 Pobieranie jednostki po identyfikatorze

Aby pobrać jednostkę należy spełnić jeden warunek podać identyfikator jednostki.

Przykładowe pobranie jednostki:

```
var res = client.Post(new PobierzJednostkęPoIdRequest()
{
    IdentyfikatorJednostki = 1
}) as PobierzJednostkęPoIdResponse;
```

### 5.21.1 DTO request – PobierzJednostkęPoIdRequest

```
[Route("/Jednostka/PoId")]
public class PobierzJednostkęPoIdRequest : RequestBaseDto,
IReturn<PobierzJednostkęPoIdResponse>
{
    public int IdentyfikatorJednostki { get; set; }
}
```

### 5.21.2 DTO response – PobierzJednostkęPoIdResponse

Zwracany jest wewnątrz obiekt JednostkaDto opisana w [6.4 JednostkaDto](#).

```
public class PobierzJednostkęPoIdResponse : ResponseBaseDto
{
    public JednostkaDto Jednostka { get; set; }
}
```

## 5.22 Pobieranie koszulki po identyfikatorze koszulki

Przykład pobierania koszulki po identyfikatorze koszulki:

```
var res = client.Post(new PobierzKoszulkePoIdRequest()
{
    Id = 91891
});
```

### 5.22.1 DTO request – PobierzKoszulkePoIdRequest

```
[Route("/Koszulka/PoId")]
public class PobierzKoszulkePoIdRequest : RequestBaseDto,
IReturn<PobierzKoszulkePoIdResponse>
{
    public virtual int Id { get; set; }
}
```

### 5.22.2 DTO response – PobierzKoszulkePoIdResponse

Zwracany jest wewnątrz obiekt `PismoDto` opisany w *PismoDto (Koszulka)6.1*

```
public class PobierzKoszulkePoIdResponse : ResponseBaseDto
{
    public PismoDto Pismo { get; set; }
}
```

## 5.23 Pobierz koszulkę po znaku sprawy

Przykład pobierania koszulki po znaku sprawy:

```
var res = client.Post(new (new PobierzKoszulkePoZnakuSprawyRequest()
{
    Znak = "BI.III.000.10.2013"
}) as PobierzKoszulkePoZnakuSprawyResponse;
```

### 5.23.1 DTO request – PobierzKoszulkePoZnakuSprawyRequest

```
[Route("/Koszulka/PobierzPoZnakuSprawy")]
public class PobierzKoszulkePoZnakuSprawyRequest : RequestBaseDto,
IReturn<PobierzKoszulkePoZnakuSprawyResponse>
{
    public virtual string Znak { get; set; }
}
```

### 5.23.2 DTO response – PobierzKoszulkePoZnakuSprawyResponse

Zwracany jest wewnątrz obiekt `PismoDto` opisany w *PismoDto (Koszulka)6.1*

```
public class PobierzKoszulkePoZnakuSprawyResponse: ResponseBaseDto
{
    public virtual PismoDto Pismo { get; set; }
}
```

## 6 Globalne Data Transfer Object

### 6.1 PismoDto (Koszulka)

```
public class PismoDto
{
    /// <summary>
    /// Identyfikator koszulki
```





```
/// </summary>
public virtual int ID { get; set; }
/// <summary>
/// Nazwa wyświetlana koszulki
/// </summary>
public virtual string Nazwa { get; set; }
/// <summary>
/// RodzajKoszulki
/// </summary>
public virtual int? Rodzaj { get; set; }
/// <summary>
///
/// </summary>
public virtual DateTime? WazneDo { get; set; }
/// <summary>
/// Data utworzenia pisma w systemie
/// </summary>
public virtual DateTime DataUtworzenia { get; set; }
/// <summary>
/// Pismo zawieszone prawda lub fałsz
/// </summary>
public virtual bool Zawieszone { get; set; }
/// <summary>
/// Pismo zakończenia
/// </summary>
public virtual bool Zakonczone { get; set; }
/// <summary>
/// Data zakończenia może być nullem
/// </summary>
public virtual DateTime? DataZakonczenia { get; set; }
/// <summary>
/// Termin pisma zakończenia może być nullem
/// </summary>
public virtual DateTime? TerminPisma { get; set; }
/// <summary>
///
/// </summary>
public virtual int? ZakonczeniePowod { get; set; }
/// <summary>
/// Czy jest to koszulka wrażliwa prawda lub fałsz
/// </summary>
public virtual bool KoszulkaWrazliwa { get; set; }
/// <summary>
///
/// </summary>
public virtual DateTime? KoszulkaWrazliwaData { get; set; }
/// <summary>
/// Czy koszulka została zarchiwizowana
/// </summary>
public virtual bool CzyZarchiwizowany { get; set; }
/// <summary>
/// Stanowisko twórcy koszulki
/// </summary>
public StanowiskoDto TworcaStanowisko { get; set; }
}
```

## 6.2 StanowiskoDto (Stanowisko pracownika)



```
public class StanowiskoDto : ResponseBaseDto
{
    /// <summary>
    /// Identyfikator stanowiska
    /// </summary>
    public int Id { get; set; }
    /// <summary>
    /// Identyfikator oryginału stanowiska (wartość nullable)
    /// </summary>
    public int? IdOriginal { get; set; }
    /// <summary>
    /// Identyfikator jednostki przyporządkowania pracownika (wartość nullable)
    /// </summary>
    public int IdJednostki { get; set; }
    /// <summary>
    /// Identyfikator jednostki oryginał jednostki przyporządkowania pracownika
    /// (wartość nullable)
    /// </summary>
    public int? IdJednostkiOriginal { get; set; }
    /// <summary>
    /// Identyfikator jednostki wydziału przyporządkowania pracownika (wartość
    nullable)
    /// </summary>
    public int? IdWydzialu { get; set; }
    /// <summary>
    /// Identyfikator wydziału jednostki oryginał przyporządkowania pracownika
    /// (wartość nullable)
    /// </summary>
    public int? IdWydzialuOriginal { get; set; }
    /// <summary>
    /// Identyfikator pracownika (wartość nullable)
    /// </summary>
    public int? IdPracownika { get; set; }
    /// <summary>
    /// Identyfikator pracownika oryginał (wartość nullable)
    /// </summary>
    public int? IdPracownikaOriginal { get; set; }
    /// <summary>
    /// Nazwa stanowiska
    /// </summary>
    public string Nazwa { get; set; }

    /// <summary>
    /// Kolejność sortowania w jednostce przyporządkowania
    /// </summary>
    public int SortOrder { get; set; }

    /// <summary>
    /// Data utworzenia pracownika (wartość nullable)
    /// </summary>
    public DateTime? DataUtworzenia { get; set; }
    /// <summary>
    /// czy pracownik jest ukryty w strukturze jednostek prawda lub fałsz
    /// </summary>
    public bool Ukryty { get; set; }
}
```

### 6.3 PracownikDto

```
public class PracownikDto : ResponseBaseDto
{
    /// <summary>
    /// Identyfikator pracownika
    /// </summary>
    public virtual int ID { get; set; }
    /// <summary>
    /// Imie pracownika
    /// </summary>
    public virtual string Imie { get; set; }
    /// <summary>
    /// Nazwisko pracownika
    /// </summary>
    public virtual string Nazwisko { get; set; }
    /// <summary>
    /// Nazwa stanowiska pracownika
    /// </summary>
    public virtual string Stanowisko { get; set; }
    /// <summary>
    /// Pole ustawień użytkownika
    /// </summary>
    public virtual string Ustawienia { get; set; }
    /// <summary>
    /// Inicjały użytkownika
    /// </summary>
    public virtual string Inicjaly { get; set; }
    /// <summary>
    /// Login ezd
    /// </summary>
    public virtual string Login { get; set; }
    /// <summary>
    /// Czy użytkownik jest aktywny
    /// </summary>
    public virtual bool Aktywny { get; set; }
    /// <summary>
    /// Użytkownik systemowy
    /// </summary>
    public virtual bool Systemowy { get; set; }
    public virtual string SystemInfo { get; set; }
    /// <summary>
    /// Login ActiveDirectory
    /// </summary>
    public virtual string ActiveDirectory { get; set; }
    /// <summary>
    /// Email użytkownika
    /// </summary>
    public virtual string Email { get; set; }
    /// <summary>
    /// Nazwa drukarki kodów kreskowych
    /// </summary>
    public virtual string DrukarkaKodow { get; set; }
    /// <summary>
    /// Originalnu identyfikator użytkownika przed zmianami
    /// </summary>
    public virtual int IdOryginal { get; set; }
    /// <summary>
    /// Kolejność sortowania w strukturze pracowników
    /// </summary>
```



```
public virtual int SortOrder { get; set; }
/// <summary>
/// Identyfikator domyślnego certyfikatu pracownika
/// </summary>
public virtual string DomyślnyCert { get; set; }
//public virtual string Certyfikat { get; set; }
/// <summary>
/// Unikalny identyfikator domenowy SID
/// </summary>
public virtual string ActiveDirectorySID { get; set; }
public virtual string ActiveDirectoryPrincipal { get; set; }
/// <summary>
/// Nazwa Wyświetlana
/// </summary>
public virtual string DisplayName { get; set; }
//public virtual string DisplayNamePL { get; set; }
//public virtual bool IsEmailValid { get; set; }
/// <summary>
/// Czy użytkownik jest ukryty w strukturze
/// </summary>
public virtual bool Ukryty { get; set; }
/// <summary>
/// Włączenie modułu integracji
/// </summary>
public virtual bool ModulIntegracjaWidoczność { get; set; }
/// <summary>
/// Identyfikator jednostki pracownika w której przebywa
/// </summary>
public virtual int IdJednostki { get; set; }
/// <summary>
/// Obiekt jednostki
/// </summary>
public virtual JednostkaDto Jednostka { get; set; }
/// <summary>
/// Identyfikator jednostki oryginalny
/// </summary>
public virtual int IdJednostkiOrginal { get; set; }
/// <summary>
/// Obiekt jednostki orginal
/// </summary>
public virtual JednostkaDto JednostkaOrginal { get; set; }
/// <summary>
/// Identyfikator wydziału
/// </summary>
public virtual int? IdWydziału { get; set; }
/// <summary>
/// Obiekt wydziału typu jednostak
/// </summary>
public virtual JednostkaDto Wydział { get; set; }
/// <summary>
/// Identyfikator Wydziału orginal
/// </summary>
public virtual int? IdWydziałuOrginal { get; set; }
/// <summary>
/// Obiekt wydziału orginal
/// </summary>
public virtual JednostkaDto WydziałOrginal { get; set; }
/// <summary>
/// Id stanowiska właściciela
/// </summary>
```



```
    public virtual int? IdStanowiska { get; set; }

    /// <summary>
    /// Id stanowiska oryginalne właściciela
    /// </summary>

    public virtual int? IdStanowiskaOrginal { get; set; }
    /// <summary>
    /// Nazwa jednostki w której przebywa
    /// </summary>

    public virtual string JednostkaNazwa { get; set; }
    /// <summary>
    /// Symbol jednostki w której przebywa
    /// </summary>
    public virtual string JednostkaSymbol { get; set; }
    public virtual int Order { get; set; }
    public virtual int? IdRoliPrzelozonej { get; set; }
    public virtual DateTime? DataUtworzenia { get; set; }
    public virtual DateTime? DataUsuniecia { get; set; }
    public virtual int? IdAplikacjiAutoryzacji { get; set; }
    public virtual int? IdInstytucjiAutoryzacji { get; set; }
    public virtual int? IdPracownikaAutoryzacji { get; set; }
    public virtual int? IdStanowiskaAutoryzacji { get; set; }
    /// <summary>
    /// Atrybuty pracownika
    /// </summary>
    public virtual string Atrybut1 { get; set; }
    public virtual string Atrybut2 { get; set; }
    public virtual string Atrybut3 { get; set; }
    public virtual string Atrybut4 { get; set; }
    public virtual string Atrybut5 { get; set; }
    public virtual string Atrybut6 { get; set; }
    /// <summary>
    /// Ważność konta może być nulle jeśli jest bezterminowe
    /// </summary>

    public virtual DateTime? WazneDo { get; set; }
    /// <summary>
    /// Data Zmiany Hasła
    /// </summary>

    public virtual DateTime? DataZmianyHasla { get; set; }
}
```



## 6.4 JednostkaDto

```
public class JednostkaDto : ResponseBaseDto
{
    /// <summary>
    /// Identyfikator jednostki
    /// </summary>
    public virtual int ID { get; set; }
    /// <summary>
    /// Nazwa jednostki
    /// </summary>
    public virtual string Nazwa { get; set; }
    /// <summary>
    /// Opis jednostki
    /// </summary>
    public virtual string Opis { get; set; }
    /// <summary>
    /// Symbol jednostki
    /// </summary>
    public virtual string Symbol { get; set; }
    /// <summary>
    /// Czy jednostka jest aktywna
    /// </summary>
    public virtual bool Aktywny { get; set; }
    /// <summary>
    /// Kolejność sortowania
    /// </summary>
    public virtual int SortOrder { get; set; }
    /// <summary>
    /// Data dodania jednostki do organizacji
    /// </summary>
    public virtual DateTime DataDodania { get; set; }
    /// <summary>
    /// Id nadrzędne jednostki jeśli jest Rootem jednostki zawsze będzie null
    /// jeśli nie to określa jaka jednostka jest nad nią
    /// </summary>
    public virtual int? IdNadrzedne { get; set; }
    public virtual int? IdPoprzedniego { get; set; }
    /// <summary>
    /// Typ jednostki Jednostka / wydział / oddział
    /// </summary>
    public virtual int IdTypJednostki { get; set; }
    /// <summary>
    /// Id wydziału
    /// </summary>
    public virtual int? IdWydzialu { get; set; }
    /// <summary>
    /// Id wydziału orginal
    /// </summary>
    public virtual int? IdWydzialuOrginal { get; set; }
    /// <summary>
    /// Dodatkowe atrybuty jednostki
    /// </summary>
    public virtual string Atrybut1 { get; set; }
    public virtual string Atrybut2 { get; set; }
    public virtual string Atrybut3 { get; set; }
}
```

## 6.5 TeczkaRwaDto

```
public class TeczkaRwaDto
{
    /// <summary>
    /// Identyfikator teczki
    /// </summary>
    public virtual int ID { get; set; }
    /// <summary>
    /// Teczki podrzędne
    /// </summary>
    public virtual List<TeczkaRwaDto> TeczkiPodrzedne { get; set; }
    /// <summary>
    /// Rocznik teczki
    /// </summary>
    public virtual int Rok { get; set; }
    /// <summary>
    /// Symbol teczki
    /// </summary>
    public virtual string Symbol { get; set; }
    /// <summary>
    /// Nazwa wyświetlana teczki
    /// </summary>
    public virtual string Nazwa { get; set; }
    /// <summary>
    /// Kategoria archiwalna
    /// </summary>
    public virtual string KategoriaArchiwalna { get; set; }
    /// <summary>
    /// Typ prowadzenia  Elektroniczna = 0, Tradycyjna = 1
    /// </summary>
    public virtual TypProwadzenia TypProwadzenia { get; set; }
    public virtual int IloscWyjatkow { get; set; }
    public virtual bool UdostepnienieWyjatek { get; set; }
    public virtual int NumerLP { get; set; }
    public virtual int? LP { get; set; }
    /// <summary>
    /// Czy teczka jest aktywna
    /// </summary>
    public virtual bool Nieaktywny { get; set; }
    /// <summary>
    /// Ilość dni na załatwienie sprawy w danej kategorii
    /// </summary>
    public virtual int? TerminDni { get; set; }
}
```

## 6.6 DokumentTypeDto

```
public class DokumentTypeDto
{
    [DataMember]
    public WskazanieDokumentuDto Identyfikator { get; set; }

    [DataMember]
    public DateTime DataUtworzenia { get; set; }
}
```

```
[DataMember]
public string Nazwa { get; set; }

[DataMember]
public string Sygnatura { get; set; }

[DataMember]
public string Tytuł { get; set; }

[DataMember]
public string DataDokumentu { get; set; }

[DataMember]
public LokalizacjaDokumentuTypeDto Lokalizacja { get; set; }

[DataMember]
public string PracownikUtworzenia { get; set; }

[DataMember]
public PodpisTypeDto[] Podpis { get; set; }

[DataMember]
public string Rodzaj { get; set; }

[DataMember]
public AtrybutObiektuTypeDto[] Atrybuty { get; set; }

[DataMember]
public DokumentSkladuDto[] Skład { get; set; }
}
```

## 6.7 Enumeratory

### 6.7.1 RodzajKoszulki

```
ZWYKLA = 0,
SPRAWA = 1,
PARTIA_ARCHIWUM = 5,
MANDATY_TBDSI = 6,
DOKUMENTY_SKLADU_ARCHIWUM = 10,
DOKUMENTY_SKLADU_ARCHIWUM_ZAKONCZONE = 11
```