



*Guia de Integração
Versão 1.7*

Segurança da comunicação

Para salvaguardar a segurança do sistema de comunicações e da plataforma API gicnet a inventore, implementou várias medidas de segurança das quais destacamos:
Sistema de autenticação da API. Apenas os nossos parceiros devidamente autorizados, terão acesso aos métodos da nossa API.
Chamadas ao sistema não autorizadas serão ignoradas e descartadas.
Comunicações protegidas por SSL: As comunicações eletrónicas estão protegidas por SSL.

Webservice

A API gicnet é um webservice do tipo RESTFUL e admite no corpo dos pedidos um objeto JSON com a seguinte estrutura:

```
{"Service":"nome do serviço","idCliente":"código do cliente","senha":"palavra passe cliente/API", parametros adicionais...}
```

Endereço da API

<https://inventore.net/ws/api/ws-api.php>

Métodos disponíveis

1. Listagem de lojas

Nome do serviço: *ListaLojas*

Descrição: Método para listar as lojas (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)

Exemplo:

```
{"Service":"ListaLojas","idCliente":"0111111","senha":"xxxxx"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response":"ok", "payload": {"Lojas":[{"idloja":"1", "descritivo":"XX - VASCO GAMA"},...]}}
```

Caso a execução tenha erros:

```
{"response":"error", "payload": {"msgerror":"Não existem lojas"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response":"ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>Lojas</i>	Listagem de Lojas	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idloja</i>	Código de loja	int(10)

<i>descritivo</i>	Designação da loja	String(80)
-------------------	--------------------	------------

Caso "response"."error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

2. Dados Loja

Nome do serviço: *LojaID*

Descrição: Método para obter os dados de uma loja (existente no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idloja</i>	Código de Loja	int(10)

Exemplo:

```
{"Service":"LojaID","idCliente":"01111111","senha":"xxxxx","idloja":"1"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response":"ok", "payload": {"descritivo":"XX - COLOMBO"}}
```

Caso a execução tenha erros:

```
{"response":"error", "payload": {"msgerror":"Loja não encontrada"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response"."ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>descritivo</i>	Designação da loja	String

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

3. Listagem de Armazéns

Nome do serviço: *ListaArmazens*

Descrição: Método para listar os armazéns (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)

Exemplo:

```
{ "Service": "ListaArmazens", "idCliente": "0111111", "senha": "xxxxx" }
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{ "response": "ok", "payload": { "Armazens": [ { "idarmazem": "1", "descritivo": "VASCO GAMA", "idloja": "1" }, ... ] } }
```

Caso a execução tenha erros:

```
{ "response": "error", "payload": { "msgerror": "Não existem armazéns" } }
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>Armazens</i>	Listagem de armazéns	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idarmazem</i>	Código de armazém	int(10)
<i>descritivo</i>	Designação do armazém	String(80)

Caso "response"."error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

4. Dados Armazém

Nome do serviço: *ArmazensID*

Descrição: Método para obter os dados de um armazém (existente no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idarmazem</i>	Código de armazém	int(10)

Exemplo:

```
{"Service":"ArmazensID","idCliente":"0111111","senha":"xxxxx", "idarmazem":"1"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response":"ok", "payload": {"descritivo":"VASCO GAMA", "idloja":"1"}}
```

Caso a execução tenha erros:

```
{"response":"error", "payload": {"msgerror":"Armazém não encontrado"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>descritivo</i>	Designação do armazém	String

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

5. Listagem de Classes (Serviços)

Nome do serviço: *ListaClassesServicos*

Descrição: Método para listar as classes associadas a serviços (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)

Exemplo:

```
{"Service": "ListaClassesServicos", "idCliente": "0111111", "senha": "xxxxx"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"ClassesServicos": [{"idclasse": "120", "descritivo": "ACADEMIA XX"}, ...]}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Não existem ClassesServicos"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>ClassesServicos</i>	Listagem de classes associadas a Serviços	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idclasse</i>	Código da classe	int(10)
<i>descritivo</i>	Designação da classe	String(80)

Caso "response"."error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

6. Dados Classe (Serviços)

Nome do serviço: *ClassesServicosID*

Descrição: Método para obter os dados de uma classe associada a serviços (existente no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idclasse</i>	Código da classe	int(10)

Exemplo:

```
{"Service":"ClassesServicosID","idCliente":"0111111","senha":"xxxxx", "idclasse":"3"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response":"ok", "payload": {"descritivo":"ACADEMIA XX"}}
```

Caso a execução tenha erros:

```
{"response":"error", "payload": {"msgerror":"ClasseServico não encontrada"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
-----------	-----------	------

<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>descritivo</i>	Designação da classe	String

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

7. Listagem de Classes (Produtos)

Nome do serviço: *ListaClassesProdutos*

Descrição: Método para listar as classes associadas a produtos (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)

Exemplo:

```
{"Service": "ListaClassesProdutos", "idCliente": "0111111", "senha": "xxxxx"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"ClassesProdutos": [{"idclasse": "53", "descritivo": "LOTEAL", ...}]}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Não existem ClassesProdutos"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>ClassesProdutos</i>	Listagem de classes associadas a Produtos	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idclasse</i>	Código da classe	int(10)
<i>descritivo</i>	Designação da classe	String(80)

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

8. Dados Classe (Produtos)

Nome do serviço: *ClassesProdutosID*

Descrição: Método para obter os dados de uma classe associada a produtos (existente no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idclasse</i>	Código da classe	int(10)

Exemplo:

```
{"Service": "ClassesProdutosID", "idCliente": "0111111", "senha": "xxxxx", "idclasse": "7"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"descritivo": "LOTEAL"}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "ClasseProduto não encontrada"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: “ok”, “error”

(2) payload:

Caso “response”: “ok”:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>descritivo</i>	Designação da classe	String

Caso “response”: “error”:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

9. Listagem de Sub Classes

Nome do serviço: *ListaSubClasses*

Descrição: Método para listar sub classes (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)

Exemplo:

```
{"Service": "ListaSubClasses", "idCliente": "0111111", "senha": "xxxxx"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:**Caso a execução seja bem sucedida:**

```
{"response": "ok", "payload": {"SubClasses": [{"idclasses": "15", "descritivo": "Styling", "idclasse": "12"}, ...]}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Não existem SubClasses"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: “ok”, “error”

(2) payload:

Caso “response”.”ok”:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>SubClasses</i>	Listagem de sub classes	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idclasses</i>	Código da sub classe	int(10)
<i>descritivo</i>	Designação da classe	String(80)
<i>idclasse</i>	Código da classe de nível superior	int(10)

Caso “response”.”error”:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

10. Dados Sub Classe

Nome do serviço: *SubClasseID*

Descrição: Método para obter os dados de uma sub classe (existente no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idclasses</i>	Código da sub classe	int(10)

Exemplo:

```
{"Service": "SubClasseID", "idCliente": "0111111", "senha": "xxxxx", "idclasses": "1"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"descricao": "TÉCNICO", "idclasse": "13"}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "SubClasse não encontrada"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>descricao</i>	Designação da classe	String
<i>idclasse</i>	Código da classe de nível superior	int(10)

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

11. Listagem de Sub Classes pertencentes a uma Classe de Nível Superior

Nome do serviço: *ListaSubClassesClasseSupID*

Descrição: Método para obter os dados de sub classes associadas a uma classe de nível superior (existente no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idclasse</i>	Código da classe de nível superior	int(10)

Exemplo:

```
{"Service": "ListaSubClassesClasseSupID", "idCliente": "0111111", "senha": "xxxxx", "idclasse": "7"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"SubClasses": [{"idclasses": "15", "descritivo": "Styling"}, ...]}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "SubClasses não encontradas para a classe superior"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>SubClasses</i>	Listagem de sub classes	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idclasses</i>	Código sub classe	int(10)
<i>descritivo</i>	Designação da classe	String(80)

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

12. Listagem de Serviços/Loja

Nome do serviço: *ListaServicosLoja*

Descrição: Método para listar serviços por loja (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)

Exemplo:

```
{"Service": "ListaServicosLoja", "idCliente": "01111111", "senha": "xxxxx"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"ServicosLoja": [{"idservico": "5", "descricao": "BRUSHING", "idclasse": "140", "idclasses": "187", "idloja": "1", "preco1": "10.9", "taxaiva": "23"}, ...]}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Não existem Serviços"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>ServicosLoja</i>	Listagem de serviços por loja	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idservico</i>	Código do serviço	int(10)
<i>descricao</i>	Designação do serviço	String(80)
<i>idclasse</i>	Código classe nível superior	int(10)
<i>idclasses</i>	Código subclasse	int(10)
<i>idloja</i>	Código loja	int(10)
<i>preco1</i>	Preço 1	Double

<i>taxaiva</i>	Taxa IVA	Double
----------------	----------	--------

Caso "response"."error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

13. Listagem de Lojas com um Serviço

Nome do serviço: *ListaServicosLojaServicoID*

Descrição: Método para listar lojas com um serviço específico (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idservico</i>	Código Serviço	int(10)

Exemplo:

```
{"Service": "ListaServicosLojaServicoID", "idCliente": "0111111", "senha": "xxxxx", "idservico": "1"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"ServicosLojaServId": [{"descritivo": "BRUSHING", "idclasse": "140", "idclasses": "187", "idloja": "1", "preco1": "10.9", "taxaiva": "23"}, ...]}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Serviço não encontrado"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response"."ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>ServicosLojaServId</i>	Listagem de lojas com serviço específico	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>descritivo</i>	Designação do serviço	String(80)
<i>idclasse</i>	Código classe nível superior	int(10)
<i>idclasses</i>	Código subclasse	int(10)
<i>idloja</i>	Código loja	int(10)
<i>preco1</i>	Preço 1	Double
<i>taxaiva</i>	Taxa IVA	Double

Caso "response"."error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

14. Listagem de Produtos/Loja

Nome do serviço: *ListaProdutosLoja*

Descrição: Método para listar produtos por loja (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)

Exemplo:

```
{"Service":"ListaProdutosLoja","idCliente":"0111111","senha":"xxxxx"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response":"ok", "payload": {"ProdutosLoja":[{"idproduto":"41","descritivo":"LUO 5.3","idclasse":"57","idclasses":"69","idloja":"1","preco1":"0","preco2":"0","taxaiva":"23"},...]}
```


Caso a execução tenha erros:

```
{ "response": "error", "payload": { "msgerror": "Não existem Produtos" } }
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>ProdutosLoja</i>	Listagem de produtos por loja	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idproduto</i>	Código produto	int(10)
<i>descritivo</i>	Designação do serviço	String(80)
<i>idclasse</i>	Código classe nível superior	int(10)
<i>idclasses</i>	Código subclasse	int(10)
<i>idloja</i>	Código loja	int(10)
<i>preco1</i>	Preço 1	Double
<i>preco2</i>	Preço 2	Double
<i>preco3</i>	Preço 3	Double
<i>taxaiva</i>	Taxa IVA	Double

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

15. Listagem de lojas com um Produto

Nome do serviço: *ListaProdutosLojaProdutoID*

Descrição: Método para listar lojas com um produto específico (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idproduto</i>	Código Produto	int(10)

Exemplo:

```
{"Service": "ListaProdutosLojaProdutoID", "idCliente": "0111111", "senha": "xxxxx", "idproduto": "1"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"ProdutosLojaProdId": [{"descritivo": "LUO 5.3", "idclasse": "57", "idclasses": "69", "idloja": "1", "preco1": "0", "preco2": "0", "preco3": "0", "tax aiva": "23"}, ...]}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Produto não encontrado"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>ProdutosLojaProdId</i>	Listagem de lojas com produto específico	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>descritivo</i>	Designação do serviço	String(80)
<i>idclasse</i>	Código classe nível superior	int(10)
<i>idclasses</i>	Código subclasse	int(10)
<i>idloja</i>	Código loja	int(10)

<i>preco1</i>	Preço 1	Double
<i>preco2</i>	Preço 2	Double
<i>preco2</i>	Preço 3	Double
<i>taxaiva</i>	Taxa IVA	Double

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

16. Dados de um Produto / Loja

Nome do serviço: *ProdutoIDLojaID*

Descrição: Método para obter os dados de um produto / loja (existente no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idproduto</i>	Código do produto	int(10)
<i>idloja</i>	Código da loja	int(10)

Exemplo:

```
{"Service": "ProdutoIDLojaID", "idCliente": "0111111", "senha": "xxxxx", "idproduto": "1", "idloja": "1"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"descritivo": "LUO 5.3", "idclasse": "57", "idclasses": "69", "preco1": "0", "preco2": "0", "preco3": "0", "taxaiva": "23"}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Produto/Loja não encontrados"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>descricao</i>	Designação do serviço	String(80)
<i>idclasse</i>	Código classe nível superior	int(10)
<i>idclasses</i>	Código subclasse	int(10)
<i>preco1</i>	Preço 1	Double
<i>preco2</i>	Preço 2	Double
<i>preco3</i>	Preço 3	Double
<i>taxaiva</i>	Taxa IVA	Double

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

17. Listagem de Stocks

Nome do serviço: *ListaProdutosStock*

Descrição: Método para listar stocks de produtos (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)

Exemplo:

```
{"Service": "ListaProdutosStock", "idCliente": "0111111", "senha": "xxxxx"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"ProdutosLojaProdId": [{"idproduto": "6", "idarmazem": "1", "stock": "0"}, ...]}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Produto não encontrado"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)

<i>payload</i>	Conteúdo da resposta	JSON Object (2)
----------------	----------------------	-----------------

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response"."ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>ProdutosStock</i>	Listagem de stocks de produtos	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idproduto</i>	Código do produto	int(10)
<i>idarmazem</i>	Código armazém	int(10)
<i>stock</i>	Stock	Double

Caso "response"."error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

18. Listagem de Stocks para um Armazém

Nome do serviço: *ListaProdutosStockArmazemID*

Descrição: Método para listar stocks para um armazém (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idarmazem</i>	Código de armazém	int(10)

Exemplo:

```
{"Service":"ListaProdutosStockArmazemID","idCliente":"0111111","senha":"xxxxx","idarmazem":"1"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"ProdutosStockArmazemID": [{"idproduto": "6", "stock": "0"}, ...]}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Não existem Stocks para este Armazém"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>ProdutosStockArmazemID</i>	Listagem de stocks para um armazém	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idproduto</i>	Código do produto	int(10)
<i>stock</i>	Stock	Double

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

19. Listagem de Stocks para um Produto

Nome do serviço: *ListaProdutosStockProdutoID*

Descrição: Método para listar stocks (por armazém) para um produto (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idproduto</i>	Código do produto	int(10)

Exemplo:

```
{"Service":"ListaProdutosStockProdutoID","idCliente":"0111111","senha":"xxxxx","idproduto":"1"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response":"ok", "payload": {"ProdutosStockProdutoID": [{"idarmazem":"1", "stock":"0"},...]}}
```

Caso a execução tenha erros:

```
{"response":"error", "payload": {"msgerror":"Não existem Stocks para este Produto"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>ProdutosStockProdutoID</i>	Listagem de stocks para um produto	JSON array (3)

(3) cada elemento da lista é do tipo JSON object e contém os seguintes campos:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idarmazem</i>	Código de armazém	int(10)
<i>stock</i>	Stock	Double

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

20. Dados de Stock Produto/Armazém

Nome do serviço: *ProdutoIDArmazemID*

Descrição: Método para obter dados de stock para um produto/armazém (existentes no gicnet)

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idproduto</i>	Código do produto	int(10)
<i>idarmazem</i>	Código do armazém	int(10)

Exemplo:

```
{"Service": "ProdutoIDArmazemID", "idCliente": "01111111", "senha": "xxxxx", "idproduto": "1", "idarmazem": "1"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"stock": "2"}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Não existe este produto/armazém"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>stock</i>	Stock	Double

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

21. Criação de Encomendas no gicnet

Nome do serviço: *GravaEncomendaInternet*

Descrição: Método para criar uma encomenda no gicnet, onde os produtos são referenciados por código de artigo (gicnet).

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>nomeCliente</i>	Nome do Cliente	String(80)
<i>morada</i>	Morada Facturação	String(80)
<i>ncontribuinte</i>	Nº Contribuinte	String(15)
<i>email</i>	E-mail	String(80)
<i>movel</i>	Nº Telemovel	String(80)
<i>cPostal</i>	Código Postal Facturação	String(80)
<i>localidade</i>	Localidade Facturação	String(80)
<i>nrencomendaext</i>	Numero Encomenda Externa	String (20)
<i>linhasEncomenda</i>	Linhas da Encomenda	(*)

(*)**Linhas Encomenda:**

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idArtigo</i>	Código Artigo	int(10)
<i>TipoArtigo</i>	Tipo de Artigo ('S' - Serviço ou 'P' - Produto)	Char(1)
<i>preco</i>	Preço Artigo	Double
<i>desconto</i>	Desconto	Double
<i>taxaiva</i>	Taxa IVA	Double
<i>qtd</i>	Quantidade	Double

Exemplo:

```
{
  "Service": "GravaEncomendaInternet",
  "idCliente": "0111111",
  "senha": "xxxxx",
  "nomeCliente": "Cliente Teste",
  "morada": "Morada teste",
  "ncontribuinte": "123456789",
  "email": "teste@tst.com",
  "movel": "911234567",
  "cPostal": "4100-100",
  "localidade": "Porto",
  "nrencomendaext": "1000002571",
  "linhasEncomenda": [
    {
      "idArtigo": "1",
      "TipoArtigo": "P",
      "preco": "50",
      "desconto": "0",
      "taxaiva": "23",
      "qtd": "1"
    }
  ]
}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"documento": "57"}}
```

Nota: Caso o cliente com o nº de contribuinte/e-mail/telemóvel indicados não exista no gicnet este é criado automaticamente.

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Falta parametrizar a encomenda online no gicnet"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>documento</i>	Numero da encomenda criada	Double

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

22. Criação de Encomendas no gicnet (Código Barras)

Nome do serviço: *GravaEncomendaInternetCodigoBarras*

Descrição: Método para criar uma encomenda no gicnet, onde os produtos são referenciados por código de barras correspondente.

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>nomeCliente</i>	Nome do Cliente	String(80)
<i>morada</i>	Morada Facturação	String(80)
<i>ncontribuinte</i>	Nº Contribuinte	String(15)
<i>email</i>	E-mail	String(80)
<i>movel</i>	Nº Telemovel	String(80)
<i>cPostal</i>	Código Postal Facturação	String(80)
<i>localidade</i>	Localidade Facturação	String(80)
<i>nrencomendaext</i>	Numero Encomenda Externa	String (20)
<i>linhasEncomenda</i>	Linhas da Encomenda	(*)

(*)*Linhas Encomenda:*

PARÂMETRO	DESCRIÇÃO	TIPO
<i>codBarras</i>	Código Barras Artigo	int
<i>TipoArtigo</i>	Tipo de Artigo ('S' - Serviço ou 'P' - Produto)	Char(1)
<i>preco</i>	Preço Artigo	Double
<i>desconto</i>	Desconto	Double
<i>taxaiva</i>	Taxa IVA	Double
<i>qtd</i>	Quantidade	Double

Exemplo:

```
{ "Service": "GravaEncomendaInternetCodigoBarras", "idCliente": "0100966_9", "senha": "12345", "nomeCliente": "Cliente teste", "morada": "Morada teste", "ncontribuinte": "111222355", "email": "teste@hotmail.com", "movel": "911234567", "cPostal": "2000-052", "localidade": "Lisboa", "nrencomendaext": "1000002571", "linhasEncomenda": [{"codBarras": "726770972285", "TipoArtigo": "P", "preco": "50", "desconto": "10", "taxaiva": "23", "qtd": "2"}, {"codBarras": "805289114550", "TipoArtigo": "P", "preco": "10", "desconto": "0", "taxaiva": "23", "qtd": "2"}]}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"documento": "57"}}
```

Nota: Caso o cliente com o nº de contribuinte/e-mail/telemóvel indicados não exista no gicnet este é criado automaticamente.

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Falta parametrizar a encomenda online no gicnet"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>documento</i>	Numero da encomenda criada	Double

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

23. Teste de Código Artigo

Nome do serviço: *TestaCodigoArtigo*

Descrição: Método para testar se um determinado código de artigo (produto ou serviço) existe no gicnet.

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>idArtigo</i>	Código Artigo	int(10)
<i>tipoArtigo</i>	Tipo de Artigo ('S' - Serviço ou 'P' - Produto)	Char(1)

Exemplo:

```
{"Service": "TestaCodigoArtigo", "idCliente": "0111111", "senha": "xxxxx", "idArtigo": "45", "tipoArtigo": "P"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:

Caso a execução seja bem sucedida:

```
{"response": "ok", "payload": {"msg": "Artigo existente no gicnet com o código: 45"}}
```

Caso a execução tenha erros:

```
{"response": "error", "payload": {"msgerror": "Não existe no gicnet o artigo: 45"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response": "ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msg</i>	Mensagem sucesso	String

Caso "response": "error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String

24. Teste de Código de Barras do Artigo

Nome do serviço: *TestaCodigoBarras*

Descrição: Método para testar se um determinado código de barras de um artigo (produto ou serviço) existe no gicnet.

PARÂMETROS DE ENTRADA:

PARÂMETRO	DESCRIÇÃO	TIPO
<i>idCliente</i>	Código de Cliente gicnet	String(9)
<i>senha</i>	Palavra passe da API	String(10)
<i>codBarras</i>	Código Barras Artigo	int
<i>tipoArtigo</i>	Tipo de Artigo ('S' - Serviço ou 'P' - Produto)	Char(1)

Exemplo:

```
{"Service":"TestaCodigoArtigo","idCliente":"0111111","senha":"xxxxx",  
codBarras":"1234567890123", "tipoArtigo":"P"}
```

PARÂMETROS DE SAÍDA:

A resposta é um JSON Object com a seguinte estrutura:

Exemplos:**Caso a execução seja bem sucedida:**

```
{"response":"ok", "payload": {"msg":"Artigo com o código de barras existe no gicnet com  
o código: 45}}
```

Caso a execução tenha erros:

```
{"response":"error", "payload": {"msgerror":"Falta criar no gicnet o artigo com o código de  
barras: 1234567890123"}}
```

PARÂMETRO	DESCRIÇÃO	TIPO
<i>response</i>	Código da resposta	String (1)
<i>payload</i>	Conteúdo da resposta	JSON Object (2)

(1) Valores possíveis: "ok", "error"

(2) payload:

Caso "response":"ok":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msg</i>	Mensagem sucesso	String

Caso "response":"error":

PARÂMETRO	DESCRIÇÃO	TIPO
<i>msgerror</i>	Mensagem de erro	String