MIT044

Especificação de Personalização

29/01/2019

**Sumário**

[1. Dados Gerais 3](#_Toc536549421)

[2. Dados da Personalização 3](#_Toc536549422)

[3. Especificação da Personalização 3](#_Toc536549423)

[a. Processo Atual 3](#_Toc536549424)

[b. Processo Proposto 3](#_Toc536549425)

[i. Parametrizações 3](#_Toc536549426)

[ii. Execução 4](#_Toc536549427)

[iii. Customizações 5](#_Toc536549428)

[4. Aprovação 7](#_Toc536549429)

# Dados Gerais

|  |  |
| --- | --- |
|  |  |
| Nome do Cliente: CIEE CENTRO DE INTEGRACAO EMPRESA ESCOLA | Código do Cliente: T0850300 |
| Nome do Projeto: | Nº do Projeto: |
| N° Contrato: | Produto: |
| Data: 29/01/2019 | Assunto: |
| Gerente/Coordenador Projeto: | Assinatura: |

# Dados da Personalização

|  |
| --- |
| **Dados da Personalização** |
| **Extra Projeto: Sim Não** |
| **Qtd. Horas:** |
| **Criticidade para Implementação (\*):  Alto Impacto  Médio Impacto  Baixo Impacto** |
| **Responsável no Cliente:** |
| **Responsável na TOTVS:** |

(\*) Alto Impacto: Não é possível implementar sem a modificação no software;

Médio Impacto: É possível implementar com entrega das modificações após a implementação;

Baixo Impacto: É possível implementar sem a modificação no software.

# Especificação da Personalização

## Processo Atual

O cadastro de convenentes define particulariedades contratuais no cálculo da Folha de Pagamentos de funcionários contratados por localidade, que serão alocados nas instalações do cliente parceiro do CIEE.

Hoje este cadastro tem sua manutenção efetuada manualmente no Protheus com base em um ficha impressa (Matriz de Cálculo) preenchida manualmente pelo departamento responsável e não possui nenhum controle de histórico de versionamento que indica as mudanças que foram efetuadas na linha do tempo deste cadastro.

## Processo Proposto

## Parametrizações

Configuração de um Web Service na arquitetura REST em uma instância específica do Protheus apontando para o mesmo RPO, Dicionário e Banco de Dados hoje já existente.

Observar a chave SECURITY da seção Socket de configuração do Web Service, para indicar autenticação de requisição.

A chave MAXSTRINGSIZE é obrigatória na seção GENERAL com valor mínimo 10 para que o REST suporte requisições com mais de 1Mb.

## Execução

O Web Service por estar na arquitetura REST tem métodos pré-definidos, sendo:

* **GET:** Utilizado para consulta de uma lista de registros ou de um registro individual, dependendo dos parâmetros recebidos;  
  + Para este método teremos três interações possíveis:  
    - Consulta se a Matriz de um Contrato/Localidade está disponível para alteração, indicando que não há nenhuma versão Em Revisão;
    - Retorna a lista de versões de uma Matriz do Contrato/Localidade indicando seu Status
    - Consulta uma versão da Matriz correspondente a um Contrato/Localidade, se não indicar a versão que se quer consultar retorna a última versão, um parâmetro indica se considera como a última versão a versão em revisão ou a versão que esteja ativa;
* **POST:** Utilizado para inclusão de uma lista de registros ou de um registro individual, dependendo dos parâmetros recebidos;  
  + Para este método teremos uma única interação possível que visa a inclusão do Contrato, Localidade e Matriz com um único Json com todos estes dados;
  + Caso o Contrato já exista no cadastro, será considerada apenas a inclusão da Localidade e Matriz;
  + Caso o Contrato e a Localidade já exista no cadastro, será considerada apenas a inclusão da Matriz;
  + Após a inclusão da Matriz será feito o registro em tabela específica da sua versão inicial para consulta através do método **GET**;
* **PUT:** Utilizado para alteração de uma lista de registros ou de um registro individual, dependendo dos parâmetros recebidos;  
  + Para este método teremos três interações possíveis:  
    - Alterações nos dados do Contrato, sem que haja nenhum registro de versionamento;
    - Alterações nos dados do Localidade, sem que haja nenhum registro de versionamento;
    - Alterações nos dados da Matriz, com registro de versionamento em status de revisão, não permitindo nenhuma outra alteração enquanto a versão em revisão não seja efetivada ou rejeitada;
* **DELETE:** Utilizado para apagar uma lista de registros ou um registro individual, dependendo dos parâmetros recebidos. Este método não será implementado, portando não estará disponível;

## Customizações

O consumo dos métodos desenvolvidos seguem os seguintes critérios:

1. O tipo de conteúdo ( Content-Types ) recebido e retornado sempre será do tipo Json (application/json );
2. No HEADER de toda requisição HTTP é obrigatório o envio do parâmetro **tenantId** composto pela Empresa e Filial do Protheus. Caso tenha sido definida autenticação de requisição, será necessário o envio do parâmetro **Authorization** com tipo de autenticação **BASIC** composta por Usuário e Senha no formato base64. A seguir um exemplo de HEADER de requisição válido:  
     
   **GET /convenentes  
   Host: myserver:8080  
   Accept: application/json  
   Authorization: BASIC YWRtaW46MTIzNDU2  
   tenantId: 10,01**  
     
   onde:  
     
   “YWRtaW46MTIz” equivale a “admin:123” em base64  
     
   10,01 equivale a Empresa “10” e Filial “01”  
     
   Para Convenetes da Filial de São Paulo informar **tenantId: 40,01**   
     
   Para Convenetes da Filial do Rio de Janeiro informar **tenantId: 50,01**
3. O código do contrato e o código da localidade utilizados nas interações com os métodos do Web Service correspondem ao id do contrato e ao id da localidade respectivamente no sistema Kairós;
4. Na URL do Web Service serão enviados os parâmetros da requisição conforme definição mais adiante, tendo fixo o nome serviço como **convenentes**, o código do contrato e o código da localidade nos parâmetros correspondenetes, segue exemplo:  
     
   **http://<host>:<porta>/convenentes?contrato=<contrato>&localidade=<localidade>[&par1=<par1>&par2=<par2>&...]**
5. No caso de algum erro na operação, será retornado um Json com o código do erro conforme a Lista de códigos de estado HTTP, no formato:  
     
   **{ “errorCode”: <Código> , “errorMessage”:”<Descrição>” }**
6. A seguir os critérios de requisição de cada método:  
   * **GET:** No corpo da URL será passado o parâmetro **TIPO** indicando o tipo de requisição sendo:  
     1. Consulta se a última Matriz do Contrato/Localidade está disponível para alteração, indicando que não está com o status de Revisão, será retornado um array com true ou false, conforme a seguir:  
          
        [ true ] ou [ false ]  
          
        Exemplo da URL de requisição:  
          
        **http://<host>:<porta>/convenentes&contrato=<contrato>&  
        localidade=<localidade>&tipo=0**
     2. Retorna a lista de versões de uma Matriz do Contrato/Localidade indicando seu Status, sendo os códigos de status:  
        + 0 = Em Revisão;
        + 1 = Ativo;
        + 2 = Inativo;  
            
          **Atenção: Podemos ter várias Versões Inativas, mas somente uma Ativa e uma em Revisão**  
          A seguir um modelo de retorno desta requisição:  
            
          [ [ "000" , 2 ] , [ "001" , 2 ] , [ "002" , 1 ] , [ "003" , 0 ] ]  
            
          Exemplo da URL de requisição:  
            
          **http://<host>:<porta>/convenentes&contrato=<contrato>& localidade=<localidade>&tipo=1**
     3. Consulta a versão de uma Matriz correspondente a um Contrato/Localidade. No Query Parameter **VERSAO** é definido a versão que se está requisitando, se não for informado a última versão será retornada.  
        O Query Parameter **EMREV** define qual á ultima versão disponível a ser retornada, se definido como true considera a versão em revisão se false a versão ativa, seu valor default sempre será “false”;  
          
        Exemplos da URL:  
          
        **http://<host>:<porta>/convenentes&contrato=<contrato>& localidade=<localidade>&tipo=2  
          
        http://<host>:<porta>/convenentes&contrato=<contrato>& localidade=<localidade>&tipo=2&versao=006  
          
        http://<host>:<porta>/convenentes&contrato=<contrato>& localidade=<localidade>&tipo=2&emrev=true**O json de retorno será o mesmo utilizado da Matriz será o mesmo enviado na inclusão ou alteração desta versão.
   * **POST:** Inclusão do Contrato/Localidade/Matriz, vai receber Json com todos os campos de Contrato/Localidade/Matriz:  
       
     O Json recebido terá a seguinte estrutura macro:  
       
     **{  
     “CONTRATO”:{...},  
     “LOCALIDADE”: {...},  
     “MATRIZ”: {...}  
     }**Exemplo da URL:  
      **http://<host>:<porta>/convenentes&contrato=<contrato>& localidade=<localidade>**O Json de retorno seguirá o padrão da Lista de códigos de estado HTTP, no formato:  
       
     **{ “Id”: <Código> , “Descricao”:”<Descrição>” }**
   * **PUT:** No corpo da URL será passado o parâmetro indicando o tipo de requisição sendo, recebe o Json no mesmo formato do método POST:  
     1. 0 = Alteração dos dados do Contrato;
     2. 1 = Alteração dos dados da Localidade;
     3. 2 = Alteração dos dados da Matriz e registro da versão com status 0 = Revisão;   
          
        O Json recebido terá a seguinte estrutura macro:  
          
        **{  
        “CONTRATO”:{...},  
        “LOCALIDADE”: {...},  
        “MATRIZ”: {...}  
        }**Exemplos da URL:  
         **http://<host>:<porta>/convenentes&contrato=<contrato>& localidade=<localidade>&tipo=0  
          
        http://<host>:<porta>/convenentes&contrato=<contrato>& localidade=<localidade>&tipo=1  
          
        http://<host>:<porta>/convenentes&contrato=<contrato>& localidade=<localidade>&tipo=2**O Json de retorno seguirá o padrão da Lista de códigos de estado HTTP, no formato:  
          
        **{ “Id”: <Código> , “Descricao”:”<Descrição>” }**

# Aprovação

|  |  |  |
| --- | --- | --- |
| **Aprovador por** | **Assinatura** | **Data** |
| <Nome e Função> |  | **\_\_\_/\_\_\_/\_\_\_** |