# HBSIS - Prova COBOL

Uma franquia Ambev teve um aumento no número de clientes devido a um evento que irá ocorrer na cidade e está com problemas para gerenciar os atendimentos e as entregas. Com isso houve a necessidade de contratar novos vendedores e agora precisa distribuir a carteira de clientes para os novos vendedores de acordo com a geolocalização do vendedor.

Para ajudar a franquia foi solicitado a você desenvolver uma aplicação para gerenciar a carteira de clientes para os vendedores.

## Requisitos Funcionais:

#### 1) O sistema deverá ter um menu com as seguintes opções:

```

01.00 - Cadastros

01.01 - Cadastro de Cliente

01.02 - Cadastro de Vendedor

02.00 - Relatórios

02.01 - Relatório de Clientes

02.02 - Relatório de Vendedores

03.00 - Executar

03.01 - Executar Distribuição de Clientes

```

#### 2) O sistema deverá ter um cadastro de Clientes com as seguintes opções: Inclusão, Alteração, Exclusão e Importação.

- A importação deverá estar preparada com o seguinte layout:

```

Código Cliente - pic 9(007);

CNPJ - pic 9(014);

Razão Social - pic x(040);

Latitude - pic s9(003)v9(008);

Longitude - pic s9(003)v9(008);

```

- O sistema não deverá permitir inserir o mesmo CNPJ para clientes diferentes;

- O sistema deverá consistir se o CNPJ informado é válido;

- O nome do arquivo de importação poderá ser variável (Escolha do Usuário);

#### 3) O sistema deverá ter um cadastro de Vendedores com as seguintes opções: Inclusão, Alteração, Exclusão e Importação.

- A importação deverá estar preparada com o seguinte layout:

```

Código Vendedor - pic 9(003);

CPF - pic 9(011);

Nome Vendedor - pic x(040);

Latitude - pic s9(003)v9(008);

Longitude - pic s9(003)v9(008);

```

- O sistema não deverá permitir inserir o mesmo CPF para Vendedores diferentes;

- O sistema deverá consistir se o CPF informado é válido;

- O nome do arquivo de importação poderá ser variável (Escolha do Usuário);

#### 4) O sistema deverá ter um relatório de clientes com alguns filtros e opções:

```

- Tipo de Ordenação - Ascendente ou Decrescente;

- Tipo de Classificação - Código Cliente ou Razão Social;

- Filtro por Código de Cliente;

- Filtro por Razão Social(Levar em consideração palavras compostas);

- Filtro por Código de Vendedor;

```

#### 5) O sistema deverá ter um relatório de vendedores com alguns filtros e opções:

```

- Tipo de Ordenação - Ascendente ou Decrescente;

- Tipo de Classificação - Código Vendedor ou Nome Vendedor;

- Filtro por Código de Vendedor;

- Filtro por Nome Vendedor(Levar em consideração palavras compostas);

```

#### 6) O sistema deverá ter um módulo para executar a distribuição da carteira de clientes para os vendedores:

- O cliente deverá ser atrelado ao vendedor mais próximo(Distância entre dois pontos);

- No final do processo deverá ser gerado um arquivo csv com os vendedores e a distância entre eles no seguinte layout:

```

- Código Cliente;

- Razão Social Cliente;

- Código Vendedor;

- Nome Vendedor;

- Distância Entre o Cliente e Vendedor(Metros);

```

## Requisitos Técnicos:

- Por padrão a HBSIS não utiliza os comandos GO TO e NEXT SENTENCE no desenvolvimento de novas rotinas, desta forma, não é permitido o mesmo para resolução deste teste.

- Para o armazenamento de dados de Clientes e Vendedores deverá ser utilizado a estrutura de arquivos indexados de acordo com o modelo à baixo:

select arq-cliente assign to disk wid-arq-cliente

organization is indexed

access mode is dynamic

record key is ...

lock mode is manual

file status is ws-resultado-acesso.

- Utilização de CopyBook para procedures utilizadas em mais de um programa como exemplo (Validação de CPF / CNPJ);

- Utilização de Linkage para comunicação entre os programas;

- Utilização de Screen Section para elaboração de Menus e Telas;

## Entregável:

Você deve entregar um conjunto de artefatos, de acordo com o nível de complexidade que achar

melhor. Durante o desenvolvimento dos requisitos funcionais efetue o commit no diretório do GitHub.

A seguir, os níveis de complexidade e seus respectivos entregáveis:

### Fácil:

- Disponibilizado no GitHub;

- Implementação dos requisitos funcionais;

- Instruções para executar;

### Médio:

- Disponibilizado no GitHub;

- Implementação dos requisitos funcionais;

- Instruções para executar;

- Métodos de abertura de arquivo (input, output, I-O) conforme a necessidade da rotina;

- Métodos de leitura e start de registros (read e start);

- Legibilidade do código fonte;

### Difícil:

- Disponibilizado no GitHub;

- Implementação dos requisitos funcionais;

- Instruções para executar;

- Métodos de abertura de arquivo (input, output, I-O) conforme a necessidade da rotina;

- Métodos de leitura e start de registros (read e start);

- Legibilidade do código fonte;

- Tratamento de file status (Exemplo: Status 39 - "A conflict has been detected between the fixed file attributes and the attributes specified for that file in the program");

- Ao exibir mensagens de alerta, erros e etc... retornar o estado original da tela antes da mensagem e retornar o cursor para aonde ele estava;